TAMPEREEN TEKNILLINEN YLIOPISTO

Tommi Saari

Tekstieditorin puheohjaus

Diplomityö

Tarkastajat: yliopistotutkija Ossi Nykänen
tutkija Anne-Maritta Tervakari

Tarkastaja ja aihe hyväksytty
luonnontieteiden tiedekuntaneuvoston
kokouksessa 6.2.2013

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Teknis-luonnontieteellinen koulutusohjelma
Tommi Saari: Tekstieditorin puheohjaus
Diplomityö, 54 sivua
Toukokuu 2013
Pääaine: Hypermedia
Tarkastajat: Yliopistotutkija Ossi Nykänen, tutkija Anne-Maritta Tervakari
Avainsanat: Puhekäyttöliittymä, graafinen käyttöliittymä, multimodaalisuus, tekstieditori, käytettävyys, käyttöliittymäsuunnittelu

Työssä tutkitaan multimodaalista käyttöliittymää, jonka osina toimivat graafinen modaliteetti ja puhemodaliteetti. Tutkimuksen kohteena on selvittää miten modaliteetit tulisi yhdistää käytettävyyden maksimoimiseksi siten, että ne tukevat toisiaan mahdollisimman hyvin. Asiaa lähestytään pääasiassa puheohjauksen näkökulmasta.

Työ jakaantuu käsitteellisen pohjan muodostamaan teoriaosaan sekä kokeelliseen osuuteen, jossa toteutettiin Sublime Text 2 -tekstieditoriin puheohjaus. Teoriaosassa käydään läpi alan kirjallisuuteen pohjautuen puheentunnistusjärjestelmiä, eri modaliteettien ominaisuuksia sekä multimodaalista käyttöliittymäsuunnittelua. Kokeellisessa osassa testataan teorian toimivuutta käytännössä tekstieditorin puheohjauksen tapauksessa. Toteutetun puheohjausratkaisun käytettävyyttä myös arvioidaan asiantuntijamenetelmin.

Multimodaalisten sovellusten suunnitteluperiaatteita ei ole vielä kattavasti tutkittu. On kuitenkin tunnettua, että hyvin toteutetut multimodaalisuudet sovellukset ovat yksimodaalisia käytettävämpiä ja saavutettavampia. Tällä hetkellä multimodaalisten sovellusten kehittäjille ei ole tarjolla standardoituja käyttöliittymäsuunnittelun ohjeita, mutta multimodaalisten sovellusten lisääntyessä tullaan saamaan lisää arvokasta tietoa aiheesta.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Science and Engineering Technology
Tommi Saari: Voice Interface in Text Editor
Master of Science Thesis, 54 pages
May 2013
Major: Hypermedia
Examiner: Senior Research Fellow Ossi Nykänen, researcher Anne-Maritta Tervakari
Keywords: Speech user interface, graphical user interface, multimodality, text editor, usability, user interface design

This work focuses on inspecting multimodal user interfaces composed of graphical and voice modalities. The aim is to investigate how to combine these modalities to maximise application usability so that the modalities support each other as well as possible. The problem is approached mainly from the point of view of voice modality.

The work is divided into theoretical and experimental parts. In theoretical part, domain literacy is used to review automatic speech recognition systems, qualities of differect modalities and multimodal user interface design. In experimental part the theory built in theoretical part is tested in practice and a voice interface for a text editor is implemented. The usability of the said interface is also evaluated.

The principles of multimodal design are not yet thoroughly investigated. Nevertheless it is well known that well designed multimodal applications offer higher usability and accessibility than unimodal ones. As of this date, there are no standardized user interface design guidelines for multimodal applications. As more and more multimodal applications are made and evaluated, valuable information is gained about the subject.

ALKUSANAT

Haluan kiittää kaikkia työn valmiiksi saattamiseen osallistuneita tahoja. Erityiset kiitokset haluan välittää yliopistotutkija Ossi Nykäselle työn ohjaamisesta sekä tutkija Anne-Maritta Tervakarille kaikesta tarjotusta tuesta.

Tampereella, 10. toukokuuta 2013

Tommi Saari

SISÄLLYS

1. Johdanto 1
2. Puhe ja puheentunnistus 3
2.1 Puheen tasot 3
2.2 Toimintaperiaatteet 4
2.2.1 Tunnistusprosessin eteneminen 5
2.2.2 Tunnistuksessa käytetyt mallit 6
2.3 Puheentunnistusjärjestelmien ominaisuuksia 7
2.4 Haasteet 8
2.5 Tunnistustarkkuus 9
3. Multimodaalinen käyttöliittymä 11
3.1 Graafinen käyttöliittymä 11
3.2 Puhe käyttöliittymäelementtinä 11
3.3 Modaliteettien yhdistäminen 13
3.4 Vahvuudet 16
3.5 Käyttöliittymäsuunnittelu 16
3.6 Suunnitteluperiaatteet 17
3.6.1 Keskustelun ohjaus 17
3.6.2 Virheenkorjaus 18
3.6.3 Sanasto 19
3.6.4 Vahvistukset 20
3.6.5 Palaute ja ohjeet 21
3.7 Yhteenveto 21
4. Käytettävyyden arviointi 23
4.1 Arviointiprosessin suunnittelu 23
4.2 Käytettävyysvaatimukset 24
4.2.1 Informaation esittämisen vaatimukset 24
4.2.2 Tiedon syötön vaatimukset 25
4.3 Nielsenin heuristiikat 25
5. Tapaustutkimusesimerkki: Google Search by Voice 27
5.1 Puhehaun aloittaminen 27
5.2 Palaute 28
5.3 Virheenkorjaus 28
5.4 Pikakomennot 29
6. Case: Tekstieditorin puheohjaus 31
6.1 Suunnitteluvaatimukset ja -tavoitteet 31
6.2 Työkalut 32
6.2.1 Käytettävissä olevat puheentunnistusjärjestelmät 32

6.2.2 Työkalujen valinta . . . 33
6.3 Toteutettu toiminnallisuus . . . 33
6.4 Tekninen toteutus . . . 34
6.5 Suunnitteluratkaisut . . . 35
6.5.1 Sanaston valinta . . . 36
6.5.2 Komennonannon tehokkuus . . . 36
6.5.3 Tilanvaihdot . . . 36
6.5.4 Tila ja palaute . . . 37
6.5.5 Läpinäkyvyys . . . 38
6.6 Toteutus . . . 39
6.6.1 Komentomoodi . . . 39
6.6.2 Sanelumoodi . . . 40
6.6.3 Virheenkorjaus . . . 40
7. Arviointi . . . 43
7.1 Arviointisuunnitelma . . . 43
7.2 Tulokset . . . 45
7.3 Multimodaalisuus ja puheen käyttökohteet . . . 47
7.4 Johtopäätökset . . . 47
7.5 Jatkokehitys . . . 48
8. Yhteenveto . . . 50
Lähteet . . . 51

LYHENTEET JA NIIDEN MÄÄRITELMÄT

API	Application programming interface, ohjelmointirajapinta
ASR	Automatic speech recognition, automaattinen puheentunnistus
CLI	Command-line interface, komentorivikäyttöliittymä
COM	Component Object Model. Microsoftin binäärirajapinta, joka mahdollista prosessien välisen kommunikoinnin.
GMA	Google Mobile App, sovellus erinäköisille mobiililaitteille.
GUI	Graphical user interface, graafinen käyttöliittymä
HMM	Hidden Markov Model, kätketty Markovin malli
SAPI	Speech application programming interface, puheohjelmointirajapinta
VUI	Voice user interface, puhekäyttöliittymä
W3C	World Wide Web Consortium
WER	Word Error Rate, sanavirhetaso
WYSIWYG	What You See Is What You Get. Viittaa ohjelmistoihin, joissa sisältö näyttää muokattaessa samanlaiselta kuin lopputulos.

1. JOHDANTO

Graafiset käyttöliittymät ovat olleet hallitsevassa asemassa viime vuosikymmenten käyttöliittymissä, sillä ne ovat tarjonneet tehokkaan tavan tiedon esittämiseen hyödyntäen ihmisen hyvää näköaistia. Pyrkimys tehokkaampaan ja joustavampaan ihmisen ja tietokoneen väliseen vuorovaikutukseen on kuitenkin nostanut mielenkiintoa myös muihin modaliteetteihin kuten puheeseen. Puhetta pidetään kaikista luonnollisimpana kommunikaatiomuotona ihmisten välillä ja tämä lähestymistapa on haluttu siirtää myös tietokoneiden kanssa toimimiseen.

Työssä toteutetaan tekstieditorin puheohjaus, jolla pyritään tukemaan editorin tehokkaampaa käyttöä. Puheohjauksen ei ole tarkoitus korvata graafista käyttöliittymää, vaan toimia sitä tukevana modaliteettina. Keskeisenä ajatuksena on tutkia ensin miten tällainen puheohjaus tulisi toteuttaa tehokkuuden ja käytettävyyden näkökulmasta. Tätä varten selvitetään aikaisempia puhesovellusten ja puheteknologian tutkimuksia sekä niiden tuloksia. Tulosten perusteella kehitetään mainittu puheohjausmoduuli ja kerrotaan tästä kehitysprosessista. Työn aihepiiri tuli ohjaajalta.

Tässä työssä tarkastellaan graafisen käyttöliittymän ja puhekäyttöliittymän sulauttamista toisiinsa multimodaaliseksi käyttöliittymäksi, joka tarjoaa parhaat puolet kummastakin modaliteetista. Tarkoituksena on selvittää, mitkä ovat kummankin modaliteetin vahvuudet ja heikkoudet ja miten niiden yhdistäminen kannattaa toteuttaa siten, että ne paikkaavat toistensa heikkouksia. Tutkimustavoitteena on selvittää, mihin tarkoituksiin käytettäväksi puhe soveltuu tekstieditorin ohjauksessa. Työssä ei ole tarkoitus tarkastella käyttöliittymiä yksimodaalisesta näkökulmasta. Puhemodaliteetti on työssä pääpainoisena tarkastelun kohteena, koska se on vielä vähemmän käytetty, tutkittu, ja edelleen kehittyvä modaliteetti. Tästä syystä suuri osa työn tutkimuksesta keskittyy nimenomaan puheeseen ja sen ominaisuuksiin. Puhe- ja graafisia käyttöliittymiä sekä niiden yhdistämistä on myös aikaisemmin tutkittu kattavasti kirjallisuudessa (katso Gustafson 2002; Lai & Yankelovich 2003; Perakakis 2011). Aikaisempi tutkimus on keskittynyt paljon keskustelu- ja dialogipohjaisiin käyttötapauksiin. Tässä työssä tarkastelemme asiaa pääasiassa tekstieditorin näkökulmasta, jossa aloite on käyttäjällä ja kommunikaatio on yksisuuntaista.

Työn sisältö on järjestetty loogisesti eteneviin lukuihin. Seuraavassa eli toisessa luvussa käsitellään puheentunnistusteknologian toimintaa sekä puheentunnistusjär-

jestelmien ominaisuuksia. Ominaisuuksilla on keskeinen merkitys järjestelmän käyttökohteisiin ja siihen millaiseen tunnistukseen se sopii. Kolmannessa luvussa tarkastellaan graafisen käyttöliittymän ja puhekäyttöliittymän sulauttamista multimodaaliseksi kokonaisuudeksi. Luvussa tarkastellaan kummankin modaliteetin vahvuuksia ja heikkouksia sekä esitetään mitä hyötyä on multimodaalisista käyttöliittymistä. Neljännessä luvussa käsitellään käyttöliittymäsuunnittelua ja esitetään siinä huomioonotettavia asioita. Viidennessä luvussa käsitellään tapauskuvauksena Google Search by Voice -teknologian kehittämistä ja sen multimodaalisen käyttöliittymäsuunnittelun ratkaisuja. Kuudennessa luvussa kerrotaan puheohjauksen lisäämisestä tekstieditoriin. Luvussa esitellään toteutettu toiminnallisuus pääpiirteittäin sekä selitetään suunnitteluratkaisut, joilla puheohjauksesta pyrittiin tekemään mahdollisimman tehokasta. Seitsemännessä luvussa arvioidaan toteutettua puheohjausmoduulia ja esitetään sille parannnusehdotuksia. Kahdeksannessa luvussa esitetään yhteenveto tutkimuksesta.

2. PUHE JA PUHEENTUNNISTUS

Puheohjauksen hyödyntämisen uskotaan merkittävästi parantavan ihmisen ja tietokoneen välistä vuorovaikutusta. Tehokkaamman käytön lisäksi puhe kasvattaa järjestelmän saavutettavuutta mahdollistamalla sen käytön myös toimintarajoitteisille. Puhetta tukevien järjestelmien määrä kasvaa jatkuvasti, mutta edelleen puhetta pidetään toisen luokan modaliteettina eivätkä käyttäjät hyödynnä mahdollisuutta puheohjaukseen kovinkaan usein. Tilanteen parantamiseksi vaaditaan teknisiä edistyksiä sekä ymmärrystä puheen parhaista hyödyntämiskohteista, jotta käyttäjille kyetään tarjoamaan tehokkaita ja virhesietoisia ratkaisuja.

2.1 Puheen tasot

Puheen käsittelyn jäsentämiseksi on käytännöllistä tarkastella puhetta eri tasoilla. Puhe voidaan jakaa hierarkisesti ainakin kahdeksaan prosessointitasoon, joita voi hyödyntää puheen mallintamisessa (Schmandt 1994). Nämä tasot ylimmästä alimpaan ovat:

Diskurssinen	Keskustelun hallinnan taso, joka käsittelee muun muassa puheenvuoron vaihtoja sekä puhujahistoriaa, jotta pronominit kuten “minä” kyetään liittämään oikeisiin henkilöihin.
Pragmaattinen	Puheen tarkoituksen selvittäminen eli miksi sanottiin mitä sanottiin.
Semanttinen	Yksittäisten sanojen sekä koko virkkeen merkityksen selvittäminen.
Syntaksinen	Kieliopin tarkastelun sekä yleisten sääntöjen taso. Käsittelee muun muassa sanojen muotoa kuten isoja ja pieniä kirjaimia sekä lukujen muotoa (auki kirjoitettu tai numeroina).
Leksikaalinen	Kielessä olevat sanat, uusien sanojen muodostus etuliitteillä, päätteillä ja sanamuodoilla.
Foneeminen	Äänteet, jotka muodostavat lausutun virkkeen.
Artikulatorinen	Ääntämyksen taso, joka keskittyy puheen muodostukseen äänihuulilla.
Akustinen	Äänen muoto ilmanpaineen vaihteluina.

Alimmat kolme tasoa akustisesta foneemiseen muodostavat puheen perustavimman luonteen mallinnuksen. Näillä tasoilla käsitellään puheen luonnetta ilmanpaineen vaihteluna, sen tuottamista äänihuulilla sekä foneemien näkökulmasta. Tässä mallissa on esitetty puhe puhujan näkökulmasta, mutta katsottaessa kuulijan näkökulmasta tulee artikulatorinen taso korvata havaintotasolla, joka esittää äänen käsittelyä kuuloelimissä ja aivoissa.

Keskimmäisillä tasoilla leksikaalisessa ja syntaksisessa puhetta käsitellään sana- ja lausetasolla. Leksikaalinen taso käsittää käytettävissä olevan sanaston, äänteiden painotukset sekä sanojen mahdolliset merkitykset. Syntaksin taso puolestaan käsittelee sitä kuinka sanat sopivat yhteen lauserakenteessa. Siinä tarkastellaan laillisia sanayhdistelmiä. Tämä sisältää sanajärjestyksen, sanojen keskinäisen suhteen sekä aikamuodon.

Ylimmät tasot semanttinen, pragmaattinen ja diskurssinen muodostavat puheen merkityksen käsittelytasot. Semanttisessa näkökulmasta tarkastellaan kuinka sanat viittaavat maailmaan ja mitä suhteita ne kuvaavat ja ennustavat. Pragmaattisella tasolla puolestaan käsitellään puhujan tarkoitusta. Miksi hän sanoi mitä sanoi? Viimeiseksi diskurssianalyysissä käsitellään puhetta siihen osallistuvien osapuolten muodostamien puheenvuorojen kautta.

Tämä puheen tasomallinnus on viitteellinen, mutta antaa kehyksen jonka varaan puhejärjestelmiä voidaan suunnitella. Puheen perustekniikat kuten signaalianalyysi keskittyvät alimpiin ja keskimmäisiin tasoihin. Puheen ymmärtäminen, semantiikka sekä merkitys toimivat puolestaan keskimmäisten ja ylimpien tasojen kanssa.

2.2 Toimintaperiaatteet

Automaattinen puheentunnistus (Automatic Speech Recognition, ASR) määritellään prosessiksi, jossa jatkuva puhesignaali kuvataan joukoksi diskreettejä elementtejä (Makhoul & Schwartz 1994). Nämä elementit voivat olla äänteitä, sanoja tai lauseita. Puheentunnistuksen päämääränä pidetään syötteenä saadun puhesignaalin muuntamista tekstimuotoiseksi lauseeksi.

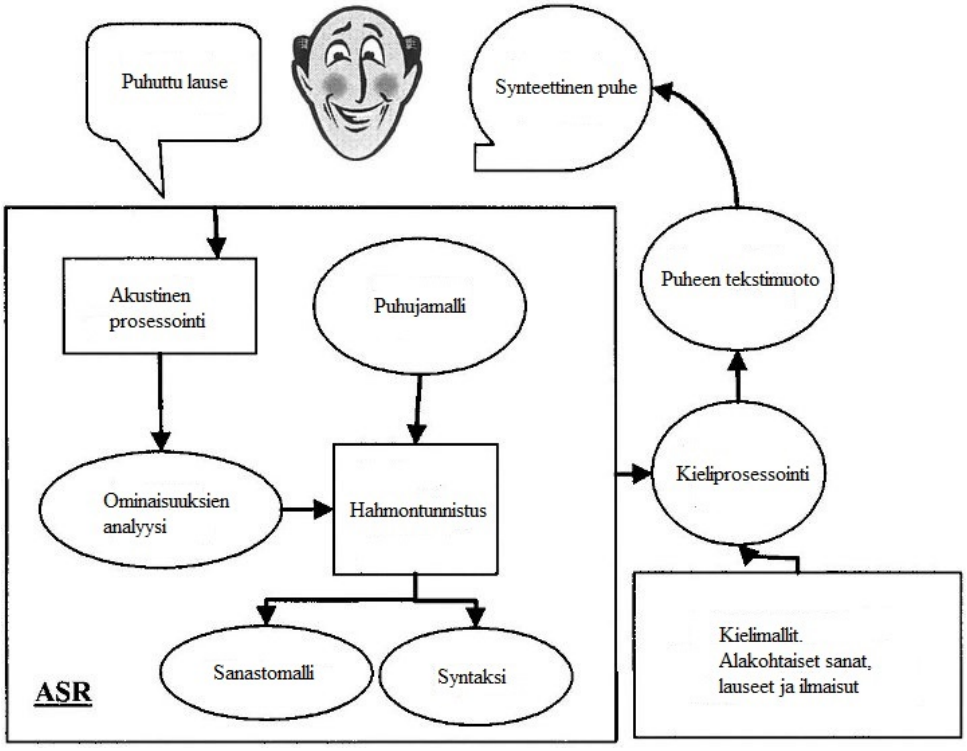
Nykyaikainen puheentunnistus perustuu tilastollisiin malleihin. Mitattua puhesignaalia verrataan olemassaoleviin malleihin ja tunnistustulokseksi valitaan teksti, jonka todennäköisyys on suurin. Toistaiseksi kuitenkin äänisignaalin ja siinä välitetyn viestin yhteyttä on mahdoton määritellä täsmällisesti, joten tunnistuksessa on jouduttu yksinkertaistamaan ongelmaa tehokkaiden matemaattisten ratkaisumenetelmien hyödyntämiseksi.

Ongelman yksinkertaistaminen perustuu sihen yleiseen ajatukseen, että puhe jakautuu hierarkisesti pienempiin komponentteihin ensin sanoiksi ja sanat edelleen foneemeiksi (äänteiksi). Tunnistuksessa pyritäänkin tunnistamaan yksittäisiä foneemeja ja yhdistämällä näitä muodostetaan koko puhuttu viesti. Foneemeillakin on

omat esiintymistodennäköisyytensä. Sanat ja lauseet muodostuvat yksinkertaisesti foneemijonoista, jonka todennäköisyys on samalla sanan tai lauseen todennäköisyys. Tämä puheentunnistuksen periaate puheen hierarkisesta jaosta on siis vain yksinkertaistettu malli eikä ole ehdottoman tarkka, mutta on käytössä, koska se tuottaa tyydyttävän tuloksen ja on matemaattisesti mallinnettavissa.

2.2.1 Tunnistusprosessin eteneminen

Chen (2006) on kuvannut lyhyesti puheentunnistusprosessin etenemisen kuvassa 2.1. Ensimmäiseksi puhesignaali vastaanotetaan mikrofonissa. Signaali muutetaan koneellisesti käsiteltävään muotoon analogisesta digitaaliseksi (akustinen prosessointi). Ominaisuuksien analyysissä puheen tärkeimmistä ominaisuuksista muodostetaan ominaisuusvektori, jossa on yleensä 10-20 piirrettä. Tämä on prosessin tärkein ja hankalin vaihe. Puhesignaali on hyvin erilainen eri puhujilla ja tilanteissa. Ominaisuuksien analyysissä pyritään karsimaan näitä häiriötekijöitä kuten taustamelua, kommunikaatiokanavan häiriöitä sekä puhujasta ja puhetavasta johtuvia vaihteluita.



Kuva 2.1: Puheentunnistuksen periaate (muokattu lähteestä Chen 2006)

Seuraava vaihe on hahmontunnistus. Ominaisuusvektoreita verrataan järjestelmään tallennettuihin malleihin sana- tai foneemitasolla. Tavoitteena on löytää täysin yhtenevä malli tai ominaisuusvektoria parhaiten vastaava. Hahmontunnistuksessa otetaan huomioon erilaiset puhenopeudet sekä kielen rakenteelliset rajoitukset. Siinä hyödynnetään akustista mallia, joka kertoo miltä sanat ja äänteet kuulostavat,

sekä sanastomallia, joka kertoo mitä sanoja on olemassa. Historiallisesti hahmontunnistuksessa on käytettä neljää eri tapaa: mallipohjiin sovitusta, sääntöihin perustuvia systeemejä, neuroverkkoja sekä kätkettyjä Markovin malleja (Hidden Markov Models, HMM). Metodit ovat vaihtoehtoisia eli niistä käytetään vain yhtä. Hahmontunnistuksen jälkeen seuraa yksinkertainen kieliprosessointivaihe, jossa signaali muutetaan tekstiksi. Kieliprosessoinnissa hyödynnetään kielimallia, joka kertoo mitä sanoja ja lauseita järjestelmä ymmärtää.

Mallipohjiin sovituksen idea on tallentaa esimerkkejä puhekuvioista. Esimerkkejä sanotaan mallipohjiksi ja ne sisältävät joukon ominaisuusvektoreita jokaista puhekuviota kohti. Tunnistuksessa saatuja ominaisuusvektoreita verrataan mallipohjiin lähimmän yhtenevyyden löytämiseksi.

Sääntöihin perustuvissa systeemeissä muodostetaan päätöspuuhun kriteerejä, joiden perusteella päätellään mitä kielen yksiköitä puhesignaaliin sisältyy. Menetelmän ongelmana on sen joustamattomuus laajaan ja monimutkaiseen puheentunnistukseen, sillä on vaikea luoda sääntöjoukkoa, joka yleistyy hyvin puhesignaalin eri variaatioille.

Neuroverkkojen ydinidea on mallintaa ihmisten aivojen toimintaa automaattisten mallien kehittämiseksi, jotka osaavat käyttäytyä älykkäästi. Neuroverkot ovat tunnettu malli tekoälyn alalla ja niiden soveltamiseen puheentunnistukseen on muutamia hyviä perusteita, keskeisimpänä ainakin niiden kyky oppia dynaamisesti esimerkeistä. Niiden käyttöön sisältyy kuitenkin puhesignaalin ajoituksiin liittyviä ongelmia, joita HMM:iä käytettäessä ei ole.

Kätketyt Markovin mallit ovat hallitsevin menetelmä hahmontunnistukseen. Niiden vahvuuksia ovat joustavuus signaalin variaatioiden käsittelyssä sekä oppivuus, sillä ne muodostavat mallinsa automaattisesti testiaineiston perusteella. Ne pystyvät hyvin ottamaan huomioon signaalin eri variaatiot ja ovat tarpeeksi voimakkaita vaativiin puheentunnistustehtäviin.

2.2.2 Tunnistuksessa käytetyt mallit

Puheentunnistuksessa käytetään yleisesti hyväksi joitain malleja, joita kustomoimalla puheentunnistusta voidaan kohdistaa eri olosuhteisiin, kielille ja sovelluskohtaisiin käyttötapauksiin. Jokainen puheentunnistusjärjestelmä sisältää nämä mallit jossain muodossa.

Akustinen malli

Hahmontunnistuksessa käytetty akustinen malli (yleensä HMM) kuvaa tilastollisesti foneemien akustiset ominaisuudet. Sen avulla lasketaan todennäköisyydet, joilla puhesignaalin segmentit (yleensä 10ms) ovat peräisin tietyistä foneemeista. Mal-

liin otetaan mukaan kaikki selvästi erotettavat foneemit, joita kielestä riippuen on eri määrä. Esimerkiksi englannin kielessä on noin 40 foneemia. Akustinen malli rakennetaan analysoimalla suuria puheaineistoja (puhekorpus) ja käyttämällä oppivia algoritmeja mallin muodostamiseen.

Kielimalli

Kieliprosessoinnissa hyödynnetään kielimallia, jonka tarkoitus on rajata sanoja järkeviksi yhdistelmiksi sovelluksen kannalta. Kielimalleja on yleisesti käytössä kahdenlaisia: laajan sanaston todennäköisyysmalleja sekä pienen sanaston kielioppeja. Laajan sanaston todennäköisyysmalli kertoo seuraavaksi odotettavissa olevan sanan kun tunnetaan N edellistä sanaa. Yleisimmin käytetään nykyisin trigrammeja, joissa huomioidaan kolme edellistä sanaa ($N = 3$). Kielimalli perustuu usein laajaan aineistoon, jossa sovelluskohtainen sanasto on oikeassa suhteessa. Mahdollisimman tarkkojen todennäköisyyksien muodostaminen on tärkeää, sillä mitä paremmin kyetään ennakoimaan seuraava sana, sitä parempi on järjestelmän puheentunnistuksen tarkkuus.

Toinen vaihtoehto kielimalliksi on kieliopin käyttö. Kielioppi on yksinkertaisempi malli, joka sisältää vain ennaltamääritellyt sanayhdistelmät, jotka puheentunnistusjärjestelmä hyväksyy. Kielimalleja käytetään komentotyyppisissä sovelluksissa, joissa on tarpeen antaa vain muutamia avainkomentoja. Järjestelmän sanavaraston laajuus voidaan jakaa kolmeen luokkaan: suppea (< 200), keskikokoinen (200 - 5000) ja laaja (> 5000). Suppean sanavaraston sovellukset toteutetaan useimmiten kieliopilla, keskikokoiset ja laajat puolestaan tilastollisella kielimallilla.

Sanastomalli

Sanastomalli sitoo edelliset mallit yhteen kertomalla mitä sanoja on olemassa ja mistä foneemeista ne muodostuvat. Suomen kielessä ääntämyksen määrittäminen on melko suoraviivaista yksikäsitteisten sääntöjen avulla, mutta esimerkiksi englannin kielessä ääntämys joudutaan määrittämään pitkälti käsin.

Mikäli sanalla on useampi ääntämys, niille on arvioitava esiintymistodennäköisyydet. Lisäksi suomen kaltaisissa morfologisesti voimakkaissa kielissä käytetään usein sanojen sijaan lyhyempiä kielen yksikköjä kuten morfeemeja. Morfeemi on kielen pienin merkityksellinen yksikkö, joita suomen kielessä ovat esimerkiksi etuliitteet, sanavartalot ja päätteet.

2.3 Puheentunnistusjärjestelmien ominaisuuksia

Puheentunnistuksen epätarkkuudesta johtuen kaikkiin tilanteisiin sopivaa tunnistinta ei ole kyetty kehittämään. Siksi on tärkeää valita tilanteeseen sopiva puheentun-

nistin, sillä sopivan järjestelmän valinnalla ennaltaehkäistään puheentunnistusvirheitä. Tämä valinta on isossa osassa puhesovellusten kehittämisessä.

Puheentunnistimen tukeman sanaston koko vaikuttaa keskeisesti järjestelmän käyttökohteisiin. Pienen sanaston tunnistimella kyetään tekemään vain komentoluonteisia sovelluksia. Keskisuurella sanastolla saadaan aikaan jo monet keskustelupohjaiset sovellukset. Laajan sanaston tunnistimet ovat taas yleisesti sanelutarkoituksiin suunnattuja.

Puheentunnistusjärjestelmät voivat olla joko puhujasta riippumattomia tai riippuvaisia. Puhujariippuvaiset järjestelmät sisältävät oppivia algoritmeja, jotka harjaantumisen kautta oppivat paremmin ymmärtämään puhujan yksilöllistä ääntämystä. Vaadittavan opetusjakson kesto voi olla muutamista minuuteista useisiin tunteihin. Pitemmällä opettamisella päästään usein myös parempiin tuloksiin. Yleensä ottaen puhujariippuvaiset järjestelmät pystyvät saavuttamaan korkeamman tunnistustarkkuuden, mutta haittapuolena ne eivät sovellu yleiseen käyttöön. Sanelusovelluksissa hyväksyttävään tarkkuuteen pääseminen voi hyvinkin edellyttää järjestelmän opettamista ymmärtämään yksittäistä käyttäjää, koska käytettävä sanasto on laaja, mikä luonnollisesti laskee tunnistustarkkuutta. Puhujariippumattomat sovellukset ovat tavoiteltavampi päämäärä, mutta tunnistusongelmien vuoksi sopivatkin vain pienen tai keskikokoisen sanaston järjestelmille.

Puhe voidaan jakaa joko yksittäiseen (isolated), epäjatkuvaan (discontinuous) tai jatkuvaan (continuous) puheeseen. Yksittäinen puhe on yksittäisiä puheentunnistusjärjestelmälle annettuja sanoja, joka tekee tunnistamisesta helppoa. Epäjatkuvassa puheessa tunnistusongelmana on tunnistaa yksittäisiä lauseita, joka on vaikeampi ongelma sanavälien tunnistamisen vuoksi. Hankalin puheentunnistuksen luokka on kuitenkin jatkuvan puheen tunnistus, jossa ongelmana on tunnistaa luonnollista sujuvaa puhetta ilman taukoja. Puheen jatkuvuudesta riippuen tunnistin soveltaa erilaisia algoritmeja, joten tunnistin on useimmiten suunnattu vain tietyntyyppisen puheen tunnistukseen.

Laajennettavuus on ominaisuus, jolla sanastoa kyetään muokkaamaan ja täydentämään käyttäjän tarpeiden mukaan. Se on hyödyllinen esimerkiksi henkilön nimien lisäämiseen sanelusovelluksessa tai omien komentojen määrittämiseen komentopohjaiseen sovelluksessa. Usein sovelluskehittäjät eivät toteuta tällaista ominaisuutta, mutta käytettävyysnäkökulmasta katsoen se olisi edullista. Komentojen muokkaaminen käyttäjän luonnollisten taipumusten tai ongelmallisten sanojen ylipääsemisen vuoksi parantaisi järjestelmän laatua.

2.4 Haasteet

Puheentunnistus on edelleen virhealtista eikä sitä ole ratkaistu tyydyttävällä tavalla laajoista yrityksistä huolimatta. Automaattisen tunnistuksen haasteita on usei-

ta. Ensinnäkin puhesignaali on luonteeltaan jatkuvaa, eikä yksittäisten sanojen ja lauseiden erottaminen siitä ole tieteellisen tarkkaa. Luonnollinen puhe ei myöskään ole sujuvaa vaan sisältää runsaasti erilaisia täyteääniä kuten epäröintejä, korjauksia ja yskintää. Näitä ei tunnistuksessa ole tarpeen huomioida, mutta niiden automaattinen erottaminen puheesta on osoittautunut hankalaksi.

Eri puhujilla esiintyy paljon vaihtelevuutta puhenopeudessa ja ääntämyksessä ja samalla puhujallakin usein eri sanoissa ja konteksteissa. Kielessä ongelmia tuottavat sellaiset sanat, jotka kuulostavat hyvin samanlaisilta (late - eight) sekä tunnistimen sanaston ulkopuoliset sanat ja varsinkin erisnimet ja vieraskieliset sanat. Sanaston ulkopuoliset sanat ovat hankala ja joskus hämmentäväkin ongelma. Mikäli lausuttu sana ei kuulu sovelluksen sanastoon, se tunnistetaan joksikin aivan toiseksi sanaksi. Näin voidaan vahingossa tulla antaneeksi komento, jota ei tarkoitettu. Ulkopuolisten sanojen havaitseminen olisi hyödyllinen ominaisuus, mutta niiden tunnistaminen on ongelmallista. Mikäli näin kyettäisiin tekemään, ratkaisu sanaston ulkopuolisiin sanoihin olisi dynaamisesti lisätä ne sanastoon ja kielioppiin niiden ilmetessä (Makhoul & Schwartz 1994).

Taustamelu hankaloittaa myös tunnistamista sekoittumalla tunnistettavaan signaaliin eikä puhetta siten kyetä erottamaan muusta melusta. Taustamelu on yksi suuria puheentunnistuksen ongelmia ja sitä hyvin sietäville järjestelmille olisi paljon kysyntää ja mikäli tällainen kyettäisiin rakentamaan, antaisi se varmasti puheentunnistusteknologialle paljon lisäarvoa ja nostaisi puheentunnistuksen suosiota. Muita mahdollisia ongelmakohtia ovat mikrofonin ja välityskanavien ominaisuudet ja ympäristön akustiikka. Esimerkiksi mikrofoniin puhuttaessa sen näytteenottotaajuuden ja bittimäärän tulisi vastata puheentunnistimen opettamisessa käytettyjä arvoja. (Kurimo 2008).

2.5 Tunnistustarkkuus

Puheentunnistuksen tarkkuus on hyvin kontekstisidonnaista. Yksikäsitteistä tarkkuutta on vaikea määritellä, sillä tarkkuuteen vaikuttavat monet asiat. Puheentunnistusteknologia on hyvin herkkä taustamelulle, joka usein estää kokonaan puhekomentojen käytön. Taustamelua vastaan voidaan taistella käyttämällä siihen sopivaa akustista mallia. Siitä huolimatta puheteknologian suurimpia haasteita onkin saada tekniikka luotettavaksi myös äänekkäissä ympäristöissä. Sanaston koko vaikuttaa myös tunnistustarkkuuteen, sillä sen suuretessa samalta kuulostavien sanojen määrä kasvaa. Pienen sanaston komentotyyppiset sovellukset voivat olla hyvinkin tarkkoja, mutta sanaston laajetessa tarkkuus heikkenee. Tämä tekee esimerkiksi tekstin sanelusta virhealtista.

Yksinkertaistettuna puheentunnistuksen suorituskyky on sen kyky tunnistaa sa-

nat oikein. Paljon käytetty mittari tarkkuuteen on WER (Word Error Rate):

$$WER = \frac{S + D + I}{N} \tag{2.1}$$

, jossa S on korvausten määrä, D on poistojen määrä ja I on lisäysten määrä jotka täytyy sanatasolla suorittaa tunnistetun lauseen muuntamiseksi referenssilauseeksi. N on referessilauseen sanojen lukumäärä. Määritelmässä mitataan väärin tulkittujen, pois jääneiden ja lisättyjen sanojen suhdetta oikeaan referenssilauseeseen. Pieni WER kertoo hyvästä tarkkuudesta.

Monet nykyiset automaattiset puheentunnistusjärjestelmät kykenevät tunnistamaan jatkuvaa puhetta ja sietävät myös yksilöllisiä puhe-eroja. Tarkkuudeksi on saatu laboratorio-olosuhteissa jopa 100% - tosin käytännön työskentely ei vastaa laboratorio-olosuhteita ja tarkkuus voi pudota merkittävästikin. Tunnistustarkkuus on kuitenkin parantunut merkittävästi viime vuosikymmeninä HMM-mallien käyttöönoton jälkeen. Viime aikoina kehitys on kuitenkin tasaantunut, koska tilastollisiin malleihin perustuva puheentunnistus on enemmän tai vähemmän saavuttanut maksimipotentiaalinsa. Suurimpia ongelmia jo puheentunnistuksen alkuajoista asti on ollut taustamelun häiritsevä vaikutus. Nykyisetkin järjestelmät ovat hyvin herkkiä ylimääräiselle äänelle. Tunnistustarkkuudet laskevat voimakkaasti meluisissa ympäristöissä. (Chen 2006).

Parhaat englanninkieliset puheentunnistimet ovat saavuttaneet jopa alle 10% WER-arvon tavallisille radio- ja televisiouutislähetyksille. Suomenkielisessä tunnistuksessa ollaan saavutettu 20% WER-arvo radiouutisissa. Koska suomen kielessä yksi sana saattaa vastata useita englanninkielisiä sanoja vahvan taivutuksen ja yhdyssanojen vuoksi, ei tunnistustarkkuus kuitenkaan ole paljoa englanninkielestä jäljessä. Suppean sanaston tunnistuksessa tarkkuudet ovat merkittävästi parempia. Tunnistin voi toimia lähes virheettömästi ja se sietää paremmin myös taustamelua sekä eri aksentteja. (Kurimo 2008).

Virheiden määrä laskee dramaattisesti yksinkertaisemmissa puheentunnistustehtävissä kuten yksittäisten sanojen tunnistuksessa. Esimerkiksi numerot 0-9 voidaan tunnistaa lähes täydellisesti 0,3%:n sanavirhetasolla.

Virheiden määrä ei suoraan korreloi järjestelmän käyttökelpoisuuteen. Tehtävästä riippuen voidaan hyväksyä eri tunnistustarkkuus. Hyvänä yleissääntönä eri tarkoituksiin voidaan pitää 90% tunnistustarkkuutta, joka on sopiva esimerkiksi sanelujärjestelmissä. Tulee kuitenkin muistaa ettei tunnistustarkkuus ole ainoa järjestelmän käyttökelpoisuuteen vaikuttava tekijä. Tähän vaikuttavat myös muun muassa järjestelmän hyödyllisyys, sosiaalinen hyväksyttävyys sekä oppimiskynnys.

3. MULTIMODAALINEN KÄYTTÖLIITTYMÄ

Multimodaaliset käyttöliittymät hyödyntävät yhtä tai useampaa modaliteettia järjestelmän syötteenä tai palautteena. Tässä työssä tarkastellaan vain graafista modaliteettia yhdistettynä puhemodaliteettiin. Näiden kahden modaliteetin on havaittu kattavasti tukevan toisiaan useimpia tarkoituksia varten (Cohen & Oviatt 1994; Grasso et al. 1998).

3.1 Graafinen käyttöliittymä

Graafiset käyttöliittymät (graphical user interface - GUI) seurasivat komentorivipohjaisia vastikkeitaan (command-line interface - CLI) ja ovat kasvaneet useimmiten käytetyksi käyttöliittymätyypiksi. Käsitteet kuten työpöytä, suora manipulointi ja WYSIWYG (What You See Is What You Get) ovat osaltaan tehneet graafisista käyttöliittymistä helpompia oppia ja käyttää kuin aikaisemmista komentorivipohjaisista vaihtoehdoistaan. Graafisia käyttöliittymiä käytetäänkin laajasti eri alustoilla kuten tietokoneissa, mobiililaitteissa ja erilaisissa automaateissa. GUI:t kehitettiin korjaamaan komentorivikäyttöliittymien korkea oppimiskynnys ja muut ongelmat. Monimutkaiset komennot voidaan laukaista helposti käyttöliittymäkomponentteja klikkaamalla. GUI:den ongelmana on mahdollisuus piilottaa toimintoja valikkohierarkioiden taakse tai epäloogisiin paikkoihin, jolloin niiden tehokkuus heikentyy. Nykyaikaisissa graafisissa käyttöliittymissä osoitinlaitteena voi toimia hiiri, kosketuskynä tai kosketusnäyttö. Tekstinsyöttölaitteena toimii fyysinen tai virtuaalinen näppäimistö tai joissakin tapauksissa käsinkirjoituksen tunnistaminen.

Tiedon esittämisessä graafiset käyttöliittymät ovat omaa luokkaansa muihin modaliteetteihin verrattuna. Graafiset käyttöliittymät ovat ulosannin luonteeltaan rinnakkaisia, sillä ne pystyvät esittämään kerralla useaa tietoa käyttäjän tarkasteltavaksi. Tässä on niiden vahvuus, sillä käyttäjä pystyy nopeasti silmäilemään graafisesta tulosteesta olennaisen tiedon.

3.2 Puhe käyttöliittymäelementtinä

Puhe ihmisten välisessä kommunikaatiossa on luonnollista, tehokasta ja suosittua. Näiden syiden takia on perusteltua soveltaa puheen käyttöä myös ihmisen ja tietokoneen väliseen kommunikointiin samojen etujen saamiseksi. Ihmisten välillä puhe

on hyvin luonnollista ja se toimiikin hallitsevana kommunikointitapana. Kuinka puhetta voi parhaiten soveltaa ihmisen ja tietokoneen väliseen kommunikointiin on perustavanlaatuinen kysymys, mutta on olemassa vahva usko siihen että tällä kyetään saavuttamaan merkittäviä etuja.

Puheen tehokkuutta puoltaa tutkimus (Chapanis 1975), jossa verrattiin eri modaliteetteja keskenään ihmisen ja tietokoneen välisessä kommunikoinnissa. Lopputuloksena selvisi, että puhe on kaikista hyödyllisin kommunikaatiomuoto ollen selvästi muita tapoja nopeampi. Huomionarvoista on myös havainto, että pelkän puheen käyttö oli lähes yhtä tehokasta kuin kaikkien muiden modaliteettien yhteiskäyttö.

Ihmisillä on myös taipumus suosia jotakin modaliteettia toisten kustannuksella. Tilanteissa, joissa kaksi tai useampi modaliteetti on keskenään yhtä tehokkaita, on havaittu että käyttäjästä riippuen jonkin modaliteetin käyttö korostuu (katso Bradford 1995; Perakakis 2011). Yleensä puhe on juuri tämä modaliteetti ja sitä voidaankin pitää suosituimpana kommunikaatiomenetelmänä. Puhekommunikaation mahdollistaminen tietokoneiden kanssa on mahdollinen keino parantaa käyttäjätyytyväisyyttä.

Puheen käyttöön on olemassa kuusi motiivia (Turunen 1998). Se voi olla ainoa, tehokkain tai miellyttävin modaliteetti ja voi toimia avustavana, vaihtoehtoisena tai korvaavana vuorovaikutuskanavana. Puheen käyttöön johtaneista motiiveista riippuen puheelle asetetaan erilaisia vaatimuksia. Ainoana modaliteettina toimiessa tunnistustarkkuuden tulee olla hyvin korkea ja käytettävissä olevan sanaston tulee olla tarkkaan mietitty. Tilanteissa, joissa puhe on valittu miellyttävyyden takia, nämä vaatimukset eivät ole yhtä korkealla käyttäjävaatimusten alenemisen johdosta.

Vastakohtana graafisille käyttöliittymille on puhe modaliteettina hyvin erityyppinen. Puhe järjestelmän ulostulona on hidasta ja vaatii käyttäjältä täydellistä keskittymistä, sillä se ei sisällä pysyvää kokoajan esillä olevaa informaatiota. Tämän sarjallisen ja tilapäisen luonteensa vuoksi se kuormittaa käyttäjän kognitiivista muistia, sillä käyttäjän täytyy muistaa ulkoa kaikki merkitsevä tieto. Puhe soveltuukin heikosti suurten tietomäärien esittämiseen. Puheulostulon käyttökohteiksi sopii paremmin käyttäjän huomion kiinnittäminen uuteen asiaan tai vaihtoehtoisen palautemekanismin tarjoaminen. Puheulostulon vahvuuksia on, että se pystytään tarjoamaan pakottamatta käyttäjää vaihtamaan kontekstia. Esimerkiksi taustalla pyörivä sähköpostiohjelma voi ilmoittaa ääneen uudesta saapuneesta sähköpostista eikä käyttäjän tarvitse avata sähköpostiohjelman ikkunaa.

Puhe on myös näkymätöntä. Käyttäjä ei voi tietää mitä komentoja ja toiminnallisuutta puhekäyttöliittymään (Voice user interface, VUI) sisältyy, ja niiden listaaminen puheen avulla olisi hyvin epäkäytännöllistä (vertaa edellä mainittu suurten tietomäärien esittäminen). Tästä seuraa, että käyttäjät koettavat antaa sellaisia komentoja joita järjestelmä ei ymmärrä ja vastavuoroisesti eivät tiedä mitä tuettuja

komentoja on käytettävissä. Sovelluksen ollessa tuntematon (uusi käyttäjä) tämä voi asettaa suuren oppimiskynnyksen järjestelmän käyttöön. Tilannetta voi verrata perinteisiin komentorivikäyttöliittymiin, jossa komennot tuli muistaa ulkoa. Puheeseen sisältyy sama ongelma, ja käyttäjän tukeminen laillisten komentojen antamisessa on erittäin tärkeää.

Vertailtaessa syötteenantoa graafiseen käyttöliittymään on puheella vahvoja käyttötapauksia. Puhekomennoilla voidaan helposti saavuttaa toiminnallisuuksia, jotka on graafisessa käyttöliittymässä piilotettu pitkän valikkohierarkian taakse. Lisäksi puheella voidaan antaa yhdellä kertaa pitempi komentosarja, joka GUI:ssa vaatisi useita hiirenpainalluksia ja välivaiheita. Puheentunnistusteknologia ei kuitenkaan muiden tunnistusteknologioiden tapaan ole täysin erehtymättömän tarkka. Puhesyöte on luonteeltaan epävarmaa erilaisten tunnistusvirheiden johdosta. Tämä aiheuttaa käyttäjälle hallinnantunteen katoamista, kun järjestelmä ei toimikaan odotetulla tavalla.

Puhekommunikaation suunta määrää pitkälti sovelluksen tyypin. Kommunikaation ollessa yksisuuntaista puhe etenee vain käyttäjältä järjestelmälle (tai toisinpäin, mitä ei tässä työssä tarkastella). Keskustelupohjaisessa kommunikaatiossa eli dialogissa kommunikaatio on molempisuuntaista. Keskustelupohjaisessa mallissa puhe on usein hallitseva modaliteetti, jota käytetään sekä syötteenantoon että palautteeseen. Tällainen käyttötapaus voisi olla esimerkiksi lomakkeen täyttö mobiililaitteen verkkoselaimessa, jossa puheohjaus on otettu käyttöön interaktion tehostamiseksi. Perinteisissä työpöydän sovelluksissa malli on taas usein yksisuuntainen, jossa aloitteen tekee käyttäjä.

3.3 Modaliteettien yhdistäminen

Puheen ja graafisen käyttöliittymän eroja on tutkittu kattavasti kirjallisuudessa. Graafinen manipulointi havaittiin tehokkaammaksi, kun “käsiteltävät objektit ovat näkyvillä, ne ovat tunnettuja eikä niitä ole liikaa valintatilanteissa” (Turunen 1998). Vastaavasti puhetta suosivat tilanteet, joissa on “tarve käsitellä objekteja, toimintoja ja tapahtumia joukosta, jota ei pystytä esittämään kerralla tai tutkimaan yksityiskohtaisesti” (Turunen 1998). Näiden vahvuuksien huomioiminen on tärkeää osa käyttöliittymäsuunnittelua. Modaliteetteja tulisi käyttää siten, että hyödynnetään niiden vahvuuksia ja käytetään toista modaliteettia korvaamaan heikot osa-alueet. Mikäli tavoitteena on täydentävä käyttöliittymä, jossa modaliteetit tukevat toisiaan eikä niillä kaikilla kyetä suorittamaan kaikkia toimintoja, tulee huomioida vastuiden oikeaoppinen jako. Korvaavassa käyttöliittymäsuunnittelussa tätä haastetta ei ole, mutta on hyvä huomioida mikä on kussakin tilanteessa sopivin modaliteetti.

Eri modaliteetteja eivät usein ole keskenään samanarvoisia. Sovelluksesta riippuen se voi olla suoramanipulointipainotteinen tai puheeseen ohjaava. Multimodaa-

liset käyttöliittymät voidaan jakaa eri luokkiin sen perusteella, mikä on sen hallitseva modaliteetti. Kummassakin tapauksessa on erilaisia hyötyjä käyttää kahta modaliteettia. Yleisemmässä tapauksessa, jossa GUI:ta täydennetään puhekomennoilla, lisähyötyä tuo käyttäjän mahdollisuus viitata toimintoihin tai objekteihin, jotka eivät ole näkyvissä. Hyödyllistä on myös mahdollisuus ilmaista toiminto ja objekti samalla kertaa ("kopioi tämä sivulle 3") sekä lisäkanava palautteen antoon puheella, joka ei tukkeuta näyttöä tai häiritse käyttäjää. (Mane et al. 1996).

Toisessa tapauksessa, jossa pääosin puhekäyttöliittymää tuetaan graafisella osalla, on myös omat hyötynsä. Graafista tulostetta voidaan käyttää kanavana, jolla käyttäjä varmistaa haluamansa toiminnon, tai keinona näyttää käytettävissä olevat puhekomennot. Järjestelmän antama puhepalaute voidaan myös tulostaa näytölle käyttäjän muistin tukemiseksi. Huomionarvoista on myös mahdollisuus valita objekteja osoittamalla niitä sen sijaan että ne valittaisiin virhealttiisti puhumalla. Objektin ollessa valittuna seuraavat puhekomennot voidaan rajata pienempään kontekstiin ottamalla huomioon vain komennot, jotka voidaan suorittaa valitulle objektille. Viimeisenä hyötynä näppäimistöä ja osoitinlaitetta voidaan käyttää varasyötteenantomekanismina mikäli puheentunnistus pettää. (Mane et al. 1996).

Multimodaalisten käyttöliittymien haasteet voidaan kiteyttää fuusio- ja fissio-ongelmiin. Modaliteettien fuusiossa ongelmana on monien eri syötteiden ja syötemuotojen yhdistäminen. Vastaavasti fissio tarkoittaa ulostulomuotojen yhdistämistä. Hyvä ratkaisu näihin ongelmiin voi parantaa merkittävästi järjestelmän käytettävyyttä tehokkuuden ja käyttäjätyytyväisyyden kasvun myötä. Usein puhutaan multimodaalisesta synergista, joka tarkoittaa käyttöliittymän suorituskyvyn paranemista yksittäisen modaliteetin versioistaan multimodaaliseen yhdistelmään. On hyvä huomata, että myös perinteinen työpöydän graafinen käyttöliittymä on luonteeltaan multimodaalinen, sillä se sisältää useimmiten näppäimistö- ja hiirisyötteen sekä antaa visuaalista palautetta. Graafiseen käyttöliittymäsuunnitteluun on olemassa valmiita suosituksia ja käytäntöjä, jotka perustuvat optimaalisen multimodaalisen synergian saavuttamiseen. Puheentunnistuksen käyttöön ei ole vielä yhtä kattavia käytäntöjä olemassa. Multimodaalista käyttöliittymää suunniteltaessa fuusio-ongelman ydin on löytää mahdollisimman hyvä sekoitus syötemodaliteetteja eli milloin pyydetään äänisyötettä ja milloin näppäimistö- ja osoitinlaitesyötettä.

Multimodaalinen syöte voidaan jakaa (W3C):

- sarjalliseen, jossa erimodaaliset syötteet tulevat peräkkäin ja vain yksi modaliteetti on kerrallaan aktiivisena.
- rinnakkaiseen, jossa voidaan samaan aikaan käsitellä useampaa toisistaan riippumatonta erimodaalista syötettä.
- yhdistettyyn, jossa voidaan samaan aikaan käsitellä useampaa toisiaan täy-

dentävää erimodaalista syötettä.

Sarjallinen multimodaliteetti on yleisin vaihtoehto ja yksinkertaisin toteuttaa. Puheentunnistuksen ja muiden uusien modaliteettien myötä myös muut vaihtoehdot ovat saaneet jalansijaa. Esimerkiksi yhdistetty syöte voi tulla kyseeseen karttaohjelmistossa, jossa käyttäjä osoittaa sormella kosketusnäytöllä haluamaansa kohdetta ja kysyy ääneen ajo-ohjeita kyseiseen sijaintiin. Käyttäjillä on luonnollinen taipumus suosia joko sarjallista tai rinnakkaista syötteenantotapaa. Tutkimuksen (Oviatt et al. 2003) mukaan käyttäjät eivät kykene vaihtamaan luonnollista taipumustaan edes siinä tapauksessa, että vaihtoehtoinen tapa olisi tehokkaampi ja vähemmän virhealtis. Käyttäjän taipumus on helppo päätellä jo lyhyestäkin aineistosta ja käyttöliittymäsuunnittelussa olisi edullista tukea tätä eikä olettaa, että käyttäjä voidaan opettaa tiettyyn toimintatapaan.

World Wide Web Consortium (W3C) myös jakaa multimodaaliset käyttöliittymät täydentäviin ja korvaaviin sen mukaan onko kaikki toiminnallisuus saavutettavissa jokaisella modaliteetilla. Korvaava käyttöliittymäsuunnittelu on toivottavaa, koska se johtaa johdonmukaiseen käyttöliittymään ja parantaa käytettävyyttä. Graafisen käyttöliittymän ja puhekäyttöliittymän sulauttamisessa saattaa kuitenkin joissakin tapauksissa olla perusteltua käyttää täydentävää ratkaisua eri modaliteettien vahvuuksien takia. Toinen modaliteetti on usein selvästi vahvempi joissakin käyttötapauksissa.

Puhekomentojen käytössä voidaan käyttää useampaa eri vuorovaikutusmoodia, jotka ilmaisevat minkä modaliteettista syötettä järjestelmä olettaa saavansa ensisijaisesti. Tällä järjestelmän kussakin tilanteessa tarjoamalla oletusmodaliteetilla on myös suora vaikutus käyttäjän modaliteetin valintaan, sillä käyttäjällä on pieni kynnys vaihtaa modaliteettia oletuksesta sopivampaan. “Klikkaa puhuaksesi” -moodi olettaa ensisijaisesti saavansa syötettä graafisen käyttöliittymän kautta, sillä se vaatii käyttäjää painamaan nappia vaihtaakseen syötettä puheeseen. “Avoin mikrofoni” -moodissa puhe puolestaan on oletusmoodi, sillä käyttäjä voi milloin tahansa antaa puhekomentoja. Näiden risteytys on “automaattinen modaliteetin valinta” -moodi, jossa järjestelmä valitsee kussakin tilanteessa jomman kumman edellä mainituista moodeista sen perusteella kumpi on oletetusti tehokkaampi. Sopivaa vuorovaikutusmoodia käyttämällä järjestelmä ilmaisee mitä modaliteettia olisi sopiva käyttää. Oikean moodin valinta palautuu fuusio-ongelmaan ja sen optimaaliseen ratkaisuun sekä multimodaalisen synergian maksimoimiseen. Käyttötapauksesta riippumatta suunnittelijan tulisi huomioida kaikkia modaliteetteja yhtälaisesti ja tarjota aina mahdollisuus vaihtaa järjestelmän tarjoama oletusmodaliteetti joksikin toiseksi.

3.4 Vahvuudet

Multimodaalisten käyttöliittymien suurimpia vahvuuksia ovat parantuneet mahdollisuudet välttää ja korjata virheitä, parantunut luotettavuus sekä vaihtoehtoiset vuorovaikutustavat. Esimerkiksi käyttäjä voi valita kirjoittavansa oudon nimen sen sijaan, että lausuisi sen. Lisäksi puheentunnistusvirheiden tapahtuessa voidaan turvautua vaihtoehtoiseen syötteenantotapaan graafiseen käyttöliittymän kautta. Käyttäjät myös pyrkivät tällaiseen toimintatapaan käyttämällä vähiten virhealtista modaliteettia (Oviatt 1999a). Lisäksi puheentunnistusvirheiden sattuessa käyttäjät pyrkivät vaihtamaan toiseen modaliteettiin (Perakakis 2011) ja näin edistämään virheestä toipumista. Multimodaalisuuden vahvuuksia ovat myös hyvät sopeutumismahdollisuudet eri tilanteisiin ja tehtäviin. Esimerkiksi julkisissa paikoissa tai meluisassa ympäristössä vältetään usein puhekomentojen käyttöä. Yksilöllisistä ominaisuuksista ja tilanne-eroista johtuen jokin modaliteetti voi olla toisia suositumpi. Käyttäjät usein myös mielellään vuorovaikuttavat monia modaliteetteja käyttäen. On tutkittu, että 56-89% käyttäjistä suosii multimodaalista vuorovaikuttamista (Cohen & Oviatt 1994). Tämä riippuu kuitenkin hyvin paljon kulloisestakin tehtävästä. Avaruudellisissa tehtävissä kuten kartan kanssa toimiessa multimodaalisuutta suosii lähes jokainen käyttäjä, mutta esimerkiksi suoritettaessa yleisiä toimintoja kuten tulostaessa dokumenttia on tavallista käyttää vain yhtä modaliteettia (Oviatt 1999b).

Usean modaliteetin käyttö voi tehdä järjestelmästä robustimman. Puheentunnistus on altis virheille, mutta graafista informaatiota hyväksikäyttäen virheitä voidaan vähentää. Keskinäinen tunnistaminen (mutual disambigation) vähentää järjestelmän virhealttiutta yhdistämällä syötteiden välittämä tieto ja päättelemällä niistä käyttäjän tarkoituksen. Puheentunnistustulosta voidaan verrata graafisen käyttöliittymän kautta saatuun syötteeseen ja tarkistaa tuloksen käypyys. Ristiriidan sattuessa tulos saatetaan jopa pystyä korjaamaan oikeaksi. (Oviatt 1999b).

Multimodaalisuus itsessään ei paranna järjestelmän käytettävyyttä. Modaliteettien oikea soveltaminen on tärkeää hyvän lopputuloksen saavuttamiseksi. Suositusten ja käytäntöjen noudattaminen on tärkeää niin multimodaalisissä kuin yksimodaalisissa käyttöliittymissä. Väärä modaliteettien käyttö voi jopa vähentää järjestelmän käytettävyyttä.

3.5 Käyttöliittymäsuunnittelu

Käyttöliittymäsuunnittelu palautuu maksimaalisen käytettävyyden saavuttamiseen. ISO 9241-11 -standardi (Ergonomics of Human System Interaction) määrittelee käytettävyyden seuraavasti: “Se vaikuttavuus, tehokkuus ja tyytyväisyys, jolla tietyt määritellyt käyttäjät saavuttavat määritellyt tavoitteet tietyssä ympäristössä”. Vai-

kuttavuus ilmaisee miten tarkasti käyttäjä saavuttaa tavoitteensa. Tehokkuus tarkoittaa käytön tehokkuutta eli miten helposti tavoitteet saavutetaan suhteessa käytettyihin resursseihin (aika, hiirenpainallukset, ym.). Tyytyväisyydellä tarkoitetaan subjektiivista käyttökokemusta eli miten tyytyväinen käyttäjä on järjestelmään riippumatta sen muista ominaisuuksista. Tätä käytettävyyden määritelmää voidaan laajentaa vielä kattamaan opittavuus, muistettavuus ja virheiden määrä (Nielsen 1993). Opittavuus kertoo kuinka helppo järjestelmän käyttö on oppia, muistettavuus ilmaisee miten helppoa vanhan käyttäjän on palauttaa mieleen järjestelmän käyttö ja virheiden määrällä tarkoitetaan käyttäjän tekemien virheiden määrää. Edellisessä johtuen suunniteltaessa multimodaalisia käyttöliittymiä hyvä käytettävyys vaatii käyttäjän psykologisten ja fyysisten ominaisuuksien huomiointia.

3.6 Suunnitteluperiaatteet

Puheteknologian rajoittuneisuus asettaa puitteet, jonka rajoissa puhekäyttöliittymiä voidaan rakentaa. Koska täysin vapaan luonnollisen kielen käyttö ei ole mahdollista, tulee puhekäyttöliittymän ohjata käyttäjää täyttämään teknologian vaatimukset (Kamm 1994). Teknologian voidaan sanoa määräävän käyttöliittymäsuunnittelua, eivätkä tehdyt ratkaisut ole käyttäjän kannalta parhaita mahdollisia. Tämä on merkittävä ongelma sovellussuunnittelijoille, mutta hyvällä suunnittelulla pystytään tässäkin tapauksessa lieventämään teknogiapainotteisten ratkaisujen vaikutusta. Turunen (1998) mainitsee työssään tärkeimpiä puhesovellusten suunnittelusääntöjä, jotka liittyvät keskustelun ohjaukseen, virheiden hallintaan, vahvistuksiin, palautteiseen ja ohjeisiin.

3.6.1 Keskustelun ohjaus

Normaalien keskustelutekniikoiden kuten vuoronvaihdon ja aiheenvaihdon hyödyntäminen ei ole nykyteknologialla helppo tehtävä. Sen sijaan keskustelyn rytmin määräävät joko käyttäjän komennot tai järjestelmän antamat kehotteet. Kehotteet ovat yleensä konseptiltaan yksinkertaisia, mutta huonosti muotoiltuna ovat virhealttiita käyttää. Käyttäjän kokemustason perusteella voidaan myös muokata kehotteita siten, että niiden tullessa tutuiksi niitä samalla lyhennetään. Mikäli kehoite annetaan puheena, huomionarvoinen asia on tukeeko järjestelmä kehoitteen keskeyttämistä eli päällepuhumista. Päällepuhuessa käyttäjä antaa vastauksensa ennenkuin kehotetta ennätetään lausumaan loppuun. Ominaisuus tehostaa järjestelmän käyttöä nopeammalla kommunikoinnilla ja lyhytkestoisen muistin rajoituksista johtuvalla virheidenehkäisyllä.

3.6.2 Virheenkorjaus

Virheiden hallinta esittää isoa roolia puhesovelluksissa ja virheiden korjaus on tärkeä osa multimodaalista käyttöliittymää. Väistämättä esiintyvien tunnistusvirheiden johdosta virheiden ennaltaehkäisy, havaitseminen, syiden etsiminen sekä korjaus ovat suunnittelussa mukaanotettavia asioita. Sovellusten tulisi luonnollisesti olla mahdollisimman virhesietoisia, jossa hyvä virheiden hallinta korostuu. Virheenkorjaukseen on monia eri vaihtoehtoja kuten uudelleen puhuminen, sanan tavutus, valitseminen vaihtoehtoisten sanojen listasta ja näppäimistöllä kirjoittaminen.

Virheiden hallinnan ensimmäinen askel, havaitseminen, on usein käyttäjän vastuulla. Mikäli järjestelmä huomaa virheen, se voi yrittää korjata sen automaattisesti, pyytää käyttäjältä vahvistusta tai vain merkitä potentiaaliset virheet käyttäjän tarkistettavaksi.

Uudelleen puhumista pidettiin aikaisemmin luontevimpana tapana korjata virheet, koska se on suosittu tapa toimia ihmisten välisessä dialogissa. Kuitenkin koneiden kanssa toimiessa lausutun uudelleen toistaminen ei paranna tunnistustarkkuutta, koska se ei korjaa perimmäistä ongelmaa eli tunnistusmallien eroavaisuutta puhujasta ja tilanteesta. Vaikutus on itse asiassa päinvastainen. Uudelleenpuhuessa tunnistustarkkuus heikkenee, koska käyttäjillä on tälloin tapana ääntää ylikorostetun selvästi, joka pahentaa akustisen mallin ongelmia (Oviatt et al. 1998). Tämä johtaa toistuviin virheisiin. Uudelleenpuhumisen ongelma virheenkorjauksessa on nimenomaan se, että tunnistustarkkuus on heikko ja heikkenee entisestään lisäyrityksillä. Puheteknologian kehittyessä on mahdollista, että siitä tulee varteenotettava vaihtoehto. Mikäli tunnistustarkkuus paranisi riittävään tarkkuuteen, uudelleen puhuminen olisi keskimäärin kirjoittamista nopeampi tapa (Suhm et al. 2001). Uudelleenpuhumisen tarkkuutta voidaan myös koettaa parantaa pienentämällä sanavarastoa. Tällöin tarkastellaan vain potentiaalisia sanoja, joiden joukossa oikean sanan ajatellaan olevan. Menetelmän heikkoutena on kuitenkin mahdollisuus, ettei oikea sana kuulu potentiaalisten sanojen joukkoon.

On tutkittu, että multimodaalinen virheenkorjaus on tehokkaampaa kuin yksimodaalinen (totta myös muille modaliteeteille kuin puheelle). Käyttäjillä on myös taipumus vaihtaa modaliteettia virheen sattuessa, mikä edesauttaa virheestä toipumista (Oviatt & VanGent 1994). Näistä johtuen graafisen käyttöliittymän ja puhekäyttöliittymän sulautumassa kannattaa virheet korjata graafisen käyttöliittymän kautta. Tunnettu virheenkorjaustapa on valita oikea sana vaihtoehtoisten sanojen listasta (niinsanottu N-Best -lista). Se on kuitenkin kyseenalainen keino, sillä usein oikea sana ei esiinny listassa ollenkaan. Eräässä tutkimuksessa (Suhm et al. 2001) havaittiin, että oikea sana oli 6 parhaan vaihtoehdon joukossa vain 24% ajasta. Näppäimistön ja hiiren yhdistelmä oli kuitenkin samassa tutkimuksessa paras yhdistelmä

virheenkorjaukseen ollen yli kaksinkertaisesti nopeampi kuin uudelleenpuhuminen. Korjaaminen yksinkertaisesti kirjoittamalla virheellinen teksti näppäimistöllä on tehokas tapa korjata puheentunnistusvirhe. GUI:n hyödyntäminen virheenkorjaukseen on havaittu tehokkaaksi myös muissa tutkimuksissa (katso Perakakis 2011).

Virheenkorjauksessa isona osana on myös virheiden havaitseminen, joka voi tapahtua käyttäjän taholta tai voi olla järjestelmän automaattisesti päättelemää. Tällä hetkellä kuitenkin on yleistä, että käyttäjä havaitsee itse virheen esimerkiksi osoittamalla sitä. Automaattinen virheen havaitseminen voi perustua tunnistustuloksen luottamuspisteisiin (confidence score), joita tunnistin liittää jokaiseen tunnistettuun sanaan. Alhainen luottamuspistemäärä voi siis potentiaalisesti merkitä virheellistä tunnistusta. Virheet voidaan havaita myös esimerkiksi kontekstin yhteydestä, mikäli tunnistustulos ei ole asiakohtaan sopiva. Kuitenkin luottamuspisteet ovat itsekin epävarmoja, joten sokea luottamus niihin voi antaa väärän tuloksen. Osittain tästä syystä johtuen automaattinen virheiden tunnistus ei kaikissa tapauksissa nopeuta virheiden korjausta. On spekuloitu, että mikäli virheet kyettäisiin tunnistamaan lähes 100% tarkkuudella, niiden automaattinen havaitseminen olisi ehdottoman hyödyllistä (Suhm et al. 2001). Automaattinen virheidenhavaitseminen ei kuitenkaan heikennä virheiden hallintaa, joten järjestelmän tuki tälle olisi suotavaa.

3.6.3 Sanasto

Sanaston koko vaikuttaa kahteen asiaan, jotka täytyy tasapainottaa keskenään. Suuri sanasto heikentää tunnistustarkkuutta toisiaan muistuttavien sanojen lisääntymisen vuoksi. Toisaalta se antaa käyttäjälle mahdollisuuden käyttää hyvinkin luonnollista ja vapaata kieltä. Sopivan tasapainon löytäminen on hankala tehtävä ja kysymystä tulisikin tarkastella kontekstisidonnaisesti seuraavien näkökulmien kautta (Chen 2006):

1. Määrittele toimialue, jonka sanasto voidaan rakentaa iteratiivisesti käyttämällä laajaa korpusta käyttäjien syötteestä.

2. Järjestelmän kommunikaatiokykyjen tulisi olla helposti käyttäjien havaittavissa.

3. Järjestelmän tulisi ohjata käyttäjiä pysymään näiden kykyjen rajojen sisällä.

4. Järjestelmän tulisi selvästi ilmaista miten korkean suorituskyvyn järjestelmällä voi saavuttaa.

5. Suunnittelijalla tulisi olla selvä ymmärrys siitä, miten paljon tunnistusvirheitä käyttäjät sietävät, miten kokeneita käyttäjiä heidän tulisi olla ratkaistakseen ongelmia tehokkaasti, sekä mikä määrä harjoittelua on hyväksyttävissä.

Monissa nykyisissä puhetta ymmärtävissä järjestelmissä sanaston koko on hyvin rajattu esimerkiksi pariin sataan sanaan. Sanaston valinta ja komentosyntaksin suunnittelu ovat tärkeässä asemassa näissä sovelluksissa. Käyttäjän pakottaminen sopeutumaan järjestelmän kieleen ei ole hyvä ratkaisu, sillä eri käyttäjillä on hyvin erilainen sopeutumiskyky. Sen sijaan käytetyn sanaston tulisi olla käyttäjän suosimaa kieltä ja mahdollistaa tehokkaan kommunikoinnin.

Sanasto olisi hyvä rajata mahdollisimman pieneksi käyttötarkoituksen kannalta. Tätä varten on tarpeen tutkia sovellus- ja alakohtaista sanastoa (Schneiderman 1992). Tämä onnistuu usein Wizard of Oz -menetelmällä, jossa seurataan käyttäjiä toteuttamassa eri käyttötapauksia. Muodostetun sanaston tulisi vastata käyttäjän mentaalimallia järjestelmän toiminnasta.

Samalta kuulostavia sanoja tulisi välttää, koska ne aiheuttavat usein virheellisiä tunnistuksia. Samalla valittujen sanojen olla mahdollisimman helposti käyttäjän opittavissa, jotta sanasto kyetään oppimaan tehokkaasti. Käyttäjillä on luonnollisestikin taipumus unohtaa sanaston osia, jolloin he saattavat ilmaista haluamansa komennon synonyymein ja toisin sanoin. Hyvä puhekäyttöliittymä ottaa tämän sanaston varioinnin huomioon ja pyrkii tarjoamaan vaihtoehtoisia ilmaisuja. Kaikenkattavaa tai kovin laajaa ilmaisujen variointia ei kuitenkaan ole tarpeen tukea muun muassa teknisistä syistä johtuen.

3.6.4 Vahvistukset

Vahvistuksilla pyritään ennakoimaan ja ehkäisemään virheitä. Niitä tulisi käyttää säästeliäästi, jotta käyttäjän työnkulku ei häiriinny liikaa. On kuitenkin joitakin tilanteita, joissa niiden käyttö on hyvin perusteltua. Vahvistukset voidaan jakaa eksplisiittisiin ja implisiittisiin vahvistuksiin (Schmandt 1994). Eksplisiittisissä vahvistuksissa käyttäjältä vaaditaan aktiivinen päätös toiminnon hyväksymiseksi tai perumiseksi. Niitä voidaan käyttää kriittisissä tilanteissa, joissa toimintoa ei ole mahdollista perua tai peruminen on hankalaa. Yksinkertaisen kyllä/ei -vastauksen saamiseksi vahvistuksen tunnistamisesta voidaan tehdä hyvin tarkka. Menetelmä kuitenkin rikkoo käyttäjän työnkulkua, joten eksplisiittisiä vahvistuksia tulisi välttää mahdollisuuksien mukaan.

Implisiittisissä vahvistuksissa käyttäjä voi ottaa passiivisen roolin. Vahvistuksessa kerrotaan käyttäjälle tulevasta toiminnosta ja tarjotaan lyhyt hetki, jonka aikana on mahdollisuus perua se. Menettely ei häiritse käyttäjää yhtä paljon kuin eksplisiittinen vahvistus, mutta ongelmana on määritellä tauolle sopiva pituus. Liian lyhyt tauko ei anna käyttäjälle kunnon mahdollisuutta toimia kun taas liian pitkä tauko voi aiheuttaa pitkän viiveen ennenkuin toiminto suoritetaan. Toinen hankaluus on määritellä sopiva tapa, jolla peruminen voidaan ilmaista.

3.6.5 Palaute ja ohjeet

Käyttäjien opastus eri tilanteissa on usein tarpeen. Näin on eritoten uusien käyttäjien kanssa. Liika opasteiden käyttö haittaa kuitenkin laajemmassa mittakaavassa sovelluksen käyttöä, kun käyttäjä on jo oppinut perusasiat. Opasteiden liittäminen kehotteisiin on varsinkin työnkulkua haittaava tekijä. Hyvä keino ratkaista ongelma on erottaa opasteet ja kehotteet, jolloin työnkulku on edelleen sujuvaa ja apua on saatavilla tarpeen mukaan. Puhepalautetta käytettäessä tulee muistaa sen rajoitukset ja pyrkiä välttämään pitkiä lauseita. Puhekäyttöliittymissä erityishuomiota vaatii puhekomentoihin liittyvä palaute. Käyttäjän tulisi aina tietää ymmärsikö järjestelmä annetun komennon ja, jos ymmärsi, mikä toiminto aiotaan suorittaa (Jones et al. 1992).

Tehokkaimman modaliteetin valinta palautteen antoon käyttäjän syötteen seurauksena on hyvin tutkittu ongelma. Yleisenä periaatteena ihmisten välisessä kommunikoinnissa on vastata toiselle samalla tapaa kuin mitä alunperinkin käytettiin. Ensimmäisissä alan tutkimuksissa 80-luvulla ihmisten ja koneiden välisessä kommunikoinnissa tultiin myös samaan tulokseen: puhekomentoihin tulisi vastata puhepalautteella. Kuitenkin hyvin nopeasti huomattiin, että puheen hitaus ja ohimenevä luonne tekee siitä hankalan käyttää käytännön tilanteissa. Puhepalautteen huomattiin hidastavan vuorovaikutusta ja jopa kasvattavan käyttäjän tekemien virheiden määrää. Visuaalinen palaute sitä vastoin on pysyvää ja nopeasti läpikäytävää. Se ei myöskään häiritse käyttäjän normaalia toimintaa toisin kuin puhepalaute, sillä käyttäjä voi tarkistaa tiedon ruudulta vasta sitä tarvittaessa. Hyvänä suosituksena voidaankin sanoa, että visuaalisen palautteen tulisi olla standardi ja puhepalautetta voidaan käyttää tukevana modaliteettina tai erikoistilanteissa. (Chen 2006).

3.7 Yhteenveto

Multimodaalinen käyttöliittymän käytettävyys kiteytyy hyvään modaliteettien yhteiskäyttöön. On tärkeää tunnistaa sovelluskohtaiset tilanteet, joihin kukin modaliteetti parhaiten soveltuu, ja tarjota tämä modaliteetti oletuksena tässä tilanteessa. Usein myös modaliteettien yhteiskäyttö tavoitteen saavuttamiseksi on paras ratkaisu ja tällöin tulee miettiä miten yhdistäminen toteutetaan. Eritoten on syytä muistaa VUI:n ongelmat palautteenannossa ja tukea tätä GUI:n kautta. Yhteiskäyttöön liittyy myös kysymys vuorovaikutusmoodin valinnasta ja hallitsevasta modaliteetista. Yleisemmässä tapauksessa GUI on hallitsevassa asemassa ja tällöin oletusvuorovaikutusmoodi myös heijastaa tätä, mutta tällöinkin voi olla perusteltua käyttää puhetta suosivaa oletusmoodia.

Suunnittelussa on myös hyvä muistaa, että useimmat ihmiset suosivat puhetta muiden modaliteettien kustannuksella. Usein puhetta pidetään siis hyvin miellyttä-

vänä modaliteettina. Tätä peilaten puheohjauksen mahdollisuuden voi tarjota myös tilanteissa, joihin se ei kenties ole paras vaihtoehto. Käytön miellyttävyys voi hyvinkin korvata pientä puutetta tehokkuudessa.

Tunnistustarkkuuden maksimointi on myös erittäin tärkeää puhesovelluksissa. Tähän voidaan vaikuttaa tunnistuksessa käytettävillä malleilla (luku 2.2.2), jotka tulee valita tarkoitukseen sopiviksi. Käyttöliittymäsuunnittelijan vastuulla on erityisten sanaston valinta, jota on syytä tarkastella luvussa 3.6.3 esitetyllä tavalla.

4. KÄYTETTÄVYYDEN ARVIOINTI

Puhesovellusten arvioinnissa voidaan pyrkiä joko löytämään suunnittelu- ja toteutusongelmia, määrittämään järjestelmän suorituskykyä tai arvioimaan järjestelmän sopivuutta tehtäväänsä ja sitä, miten hyvin se vastaa käyttäjien tarpeita. Arviointi voi olla luonteeltaan kvantitatiivista, jolloin tutkimus on laskennallinen ja usein tilastoihin perustuva. Tällöin pyritään usein esittämään järjestelmän ominaisuuksia lukuina tai prosentteina. Kvalitatiivisessa arvioinnissa puolestaan arvioidaan järjestelmää laadullisesti joitakin standardeita ja sääntöjä vastaan. Tällöin saadaan usein syvempää tietoa aiheesta ja löydetään myös ongelmien syitä, jolloin niiden korjaaminen on yksinkertaisempaa.

Puhekäyttöliittymien käytettävyydessä ei ole kyetty vielä muodostamaan yleisesti hyväksyttyjä tekijöitä, joiden perusteella niitä tulisi arvioida. Perinteisiä graafisten käyttöliittymien arviointimetodeja voidaan kuitenkin soveltaa myös puhekäyttöliittymiin tietyin huomioin (Farinazzo et al. 2010). Perimmäinen ero modaliteettien välillä on pysyvyys. Visuaalinen informaatio on aina näkyvillä, mutta puhe kuullaan vain kerran ja on siten ohimenevää. Arvioinnissa tuleekin kiinnittää huomiota tähän eroavaisuuteen ja muokata metodeja tilanteeseen sopivaksi.

4.1 Arviointiprosessin suunnittelu

Käytettävyysarviointi on aina monimutkaista eikä voida esittää yksikäsitteisiä tekijöitä, joita arvioida. Dybkjær & Bernsen (2001) ovat kuitenkin kehittäneet yleisen tason ohjeet, joiden avulla ongelmaa voidaan lähestyä. Ennen arvioinnin suorittamista tulee selvittää seuraavat asiat:

- Mitä arvioidaan (esimerkiksi palautetta tai virheidenhallintaa)?
- Mitä järjestelmän osaa arvioidaan (esimerkiksi tekstin sanelua)?
- Mikä on arvioinnin tyyppi (kvalitatiivinen, kvantitatiivinen)?
- Mitä ongelmakohtia etsitään (esimerkiksi ohjeiden laatua)?
- Mikä on arvioinnin tärkeys (esimerkiksi suuri tai vähäinen)?
- Miten vaikea arviointi on suorittaa?

- Mitkä ovat arvioinnin kustannukset?
- Mitä työkaluja arvioinnissa käytetään?

Listan selvittämisen tarkoituksena on tarjota arvioijalle työkehys, jonka varassa puhekäyttöliittymää voidaan arvioida tehokkaasti.

4.2 Käytettävyysvaatimukset

Yleisesti ottaen puhesovellusten tulee toteuttaa samat käyttäjävaatimukset kuin muidenkin vuorovaikutteisten järjestelmien. Käyttöliittymävaatimukset voidaan jaka kahteen luokkaan: toiminnallisiin ja ei-toiminnallisiin. Toiminnalliset vaatimukset määrittävät järjestelmän syötteet, palautteet ja niiden keskinäiset suhteet. Ei-toiminnalliset vaatimukset puolestaan määrittävät järjestelmän ominaisuudet kuten suorituskyvyn, käytettävyyden, turvallisuuden, luotettavuuden ja saatavuuden.

Tässä työssä arvioimme vain puhekäyttöliittymille ominaisia ei-toiminnallisia vaatimuksia. Salvador et al. (2008) mukaan nämä vaatimukset voidaan jakaa vielä kahteen aliluokkaan informaation esittämisen vaatimuksiin sekä tiedon syötön vaatimuksiin.

4.2.1 Informaation esittämisen vaatimukset

Palaute on tärkeää kaikessa kommunikoinnissa. Puhekäyttöliittymissä se on ensiarvoista, koska se kertoo käyttäjälle tavoitteiden edistymisestä (tai sen puutteesta). Palautetta voidaan antaa laitteiston tasolla, jolloin ilmaistaan ymmärrettiinkö käyttäjän puhe vai ei. Sekvenssitason palaute puolestaan kertoo ymmärrettiinkö puhe komennoksi ja, jos näin on, mistä komennosta on kyse. Lopuksi toimnnallisen tason palaute kertoo toiminnon edistymisestä (“Odota hetki...”).

Erilaiset ja eritasoiset käyttäjät tulisi myös ottaa huomioon. Käyttäjä tulisi tunnistaa ja käyttöliittymän sisältö ja esitystapa tulee muokata käyttäjäprofiilin mukaisesti. Esimerkiksi uutta käyttäjää voidaan opastaa yksityiskohtaisemmin kuin kokenutta käyttäjää.

Johdonmukaisuutta pidetään yhtenä tärkeimmistä käyttöliittymän ominaisuuksista. Käyttöliittymän toiminnan tulisi olla ennakoitavissa ja yhtenäistä läpi koko sovelluksen. Mielenkiintoinen ja ristiriitainen huomio on, että dynaamisesti käyttäjään mukautuvat järjestelmät menettävät johdonmukaisuuttaan mukautumisensa seurauksena.

Järjestelmän tulisi käyttää käyttäjälle tuttuja ja luontaisia sanoja. Järjestelmän kielen vastatessa käyttäjän maailmankuvaa tulkitaan järjestelmä heti paljon intuitiivisemmaksi ja positiivisemmaksi.

Käyttöliittymän tulisi minimoida käyttäjän muistikuorma. Tätä voidaan tukea esimerkiksi lyhyillä opasteilla, joilla ilmaistaan mitä käyttäjältä odotetaan seuraavaksi.

(Salvador et al. 2008).

4.2.2 Tiedon syötön vaatimukset

Ohje sovelluksen hallintaan tulisi olla aina tarvittaessa saatavilla. Ohjeita voidaan antaa myös automaattisesti, kun havaitaan käyttäjän tarvitsevan niitä. Esimerkiksi mikäli käyttäjä on pitkään hiljaa puhesyötettä odotettaessa, voidaan tukea käyttäjää kertomalla mahdolliset vaihtoehdot.

Virheenehkäisy on suositeltava käyttöliittymän ominaisuus, jotta käyttäjä ei toimi siten, että joudutaan esittämään virheviesti. Usein hyvä tapa on siirtää aloite toiminnasta järjestelmälle, kun havaitaan käyttäjän olevan vaikeuksissa.

Hyvässä käyttöliittymässä virheiden korjaus on nopeaa ja helppoa. Tämä parantaa käyttäjien tuottavuutta ja kannustaa tutkimaan järjestelmään pidemmälle. Virheidenhallinan lähestymistapoja on esimerkiksi vaihtaa syötemodaliteettia tunnistamattoman komennon vastaanottamisen jälkeen tai hyödyntää varmistusdialogeja epäselvissä tapauksissa.

(Salvador et al. 2008).

4.3 Nielsenin heuristiikat

Heuristisessa arvioinnissa käytettävyyttä arvioidaan ilman käyttäjää muisti- tai tarkistuslistaan tukeutuen. Arviointimetodilla löydetään yleisiä käytettävyysongelmia kuten epäjohdonmukaisuuksia käyttöliittymässä. Metodi on helppo ja nopea, mutta sen ongelmana voidaan pitää pintapuolisuutta, sillä syvimpien käytettävyysongelmien löytäminen vaatii usein käyttäjien mukaanottamista arviointiprosessiin. Sillä ei myöskään löydetä toiminnallisia ongelmia kuten järjestelmän hyödyllisyyttä, ominaisuuksien kattavuutta tai sopivuutta tehtävään.

Nielsenin kymmenen heuristiikan lista on tunnettu ja paljon käytetty käyttöliittymäarvioinnin alalla. Se on hyvin yleiskäyttöinen ja sopii moneen tilanteeseen. Heuristiikat sopivat myös hyvin edellä esiteltyjen puhekäyttöliittymien käytettävyysvaatimusten arviointiin. Heuristiikat ovat (Nielsen 1993):

Järjestelmän tilan näkyvyys Järjestelmän tulee aina tiedottaa mitä se on tekemässä. Palautetta tulee antaa kohtuullisessa ajassa.

Järjestelmän tulee vastata ulkomaailmaa Järjestelmän tulee puhua käyttäjän kieltä. Sanojen, ilmaisujen ja käsitteiden tulee olla käyttäjille tuttuja. Sen tulee noudattaa ympäröivän maailman sääntöjä ja esittää informaatio luonnollisessa ja loogisessa järjestyksessä.

Käyttäjän vapaus ja hallinnan tunne Käyttäjille tulee tarjota helppo poispääsytapa jokaisesta tilanteesta. Komentoja tulee voida perua ja toistaa. Käyttäjää ei saa rajoittaa tarpeettomasti.

Johdonmukaisuus ja standardit Käyttäjien ei tulisi joutua pohtimaan tarkoittavatko eri sanat, tilanteet tai toiminnot samaa asiaa.

Virheiden välttäminen Virheiden välttämistä tulee suosia virheiden korjauksen sijaan. Virhealttiita tilanteita tulee välttää tai toiminto tulee varmistaa ennen sen toteuttamista.

Tunnistamista, ei muistamista Käyttäjän muistikuorma tulee minimoida. Käyttäjälle tulee tarjota selkeät, tunnistettavat vaihtoehdot. Informaatiota ei tulisi joutua muistelemaan ohjelman osasta toiseen.

Käytön tehokkuus ja joustavuus Kokeneilla käyttäjille tulee tarjota oikoteitä. Käyttäjän tarpeiden muutoksiin tulee mukautua.

Esteettisyys ja minimalistinen suunnittelu Vain välttämätön tieto tulee tarjota. Monimutkaisuutta tulee välttää ja käyttäjää ei tule häiritä tarpeettomasti.

Auta käyttäjää tunnistamaan ja korjaamaan virheet Virheviestien tulee olla selkokielisesti ilmaistuja. Niistä tulee käydä ilmi itse ongelman sekä mahdolliset ratkaisuvaihtoehdot.

Ohjeet ja dokumentointi Järjestelmän käytön tulisi sujua ilman ohjettakin. Usein kuitenkin tämä on mahdotonta. Tällöin ohjeen tulisi olla selkeä ja siitä täytyy pystyä helposti löytämään oleellinen tieto.

Vaikka Nielsenin heuristiikkoja usein sovelletaan graafisen käyttöliittymän arviontiin, voidaan niitä niiden yleisyytensä vuoksi soveltaa myös puhekäyttöliittymiin. Käytettävyyttä arvioidessa tulee kuitenkin muistaa käyttöliittymätyyppien erot ja pohtia, miten heuristiikkoja tulee soveltaa kuhunkin tapaukseen.

5. TAPAUSTUTKIMUSESIMERKKI: GOOGLE SEARCH BY VOICE

Schalkwyk et al. (2010) ovat kuvanneet Google Search by Voice -teknologian kehittämistä sekä siinä kohdattuja ja ratkaistuja ongelmia. Se on mobiililaitteille suunnattu järjestelmä, jolla voidaan suorittaa Google-hakuja suoraan laitteelle puhumalla. Sovelluksen ensiesiintyminen tapahtui vuonna 2008, kun iPhonelle julkaistiin Google Mobile App (GMA), joka sisälsi Search by Voice -ominaisuuden. Search by Voice on saatavilla myös Android, BlackBerry ja Nokia S60 -laitteille. Nykyisin puhehaku on mahdollista myös yleisesti `google.com`-osoitteessa esimerkiksi pöytäkoneille.

Yksi Googlen tavoitteista on tehdä puheohjauksesta universaalisesti mahdollista. Tämän saavuttamiseksi vaaditaan kahta asiaa: saatavuutta sekä suorituskykyä. Saatavuus merkitsee, että puhesyöte tai -palaute on mahdollista jokaisessa vuorovaikutustilanteessa. Suorituskykyvaatimus puolestaan vaatii, että puhe toimii niin hyvin, ettei sen käyttö hidasta toimintaa. Tätä taustaa vasten Google kehitti Google Search by Voice -teknologian. Sen tavoitteena on kunnianhimoisesti tunnistaa mikä tahansa puhuttu hakukysely.

Mobiililaitteilla tapahtuva hakukyselyjen puheentunnistus on haastava tehtävä, koska syöte on arvaamatonta ja ympäristön ääniolosuhteet vaihtelevat suuresti. Lisäksi tavoite tunnistaa mikä tahansa kysely merkitsee todella laajaa sanastoa, joka siten vaatii paljon suorituskykyä. Googlen ratkaisu näihin ongelmiin on käyttää pilvilaskentaa, jonka avulla kyetään prosessoimaan enemmän dataa ja mallintamaan useampia olosuhteita kuin kenties koskaan ennen puheentunnistuksen historiassa.

Google Search by Voicen käyttöliittymä on sekoitus puhetta ja graafisia elementtejä. Googlella myös havahduttiin tähän ja ymmärrettiin, että puhetta ja graafisia elementtejä yhdistävät multimodaaliset käyttöliittymät ovat hyvin uusi ilmiö. Niihin liittyy monia vielä osin tutkimattomia haasteita ja mahdollisuuksia, joita käyttöliittymäsuunnittelussa joudutaan ratkomaan.

5.1 Puhehaun aloittaminen

Yksi suunnittelussa noussut ongelma oli päättää miten puhehaut voidaan aloittaa. Yksinkertaiselta kuulostavaan asiaan on todellisuudessa lukuisia vaihtoehtoja. Puhehaut voidaan esimerkiksi laukaista napinpainalluksella tai kiihtyvyysanturin havaitsemalla eleellä. Nappi voi olla puhelimen fyysinen nappi tai osa graafista käyt-

töliittymää. Painalluksia voi olla yksi tai kaksi tai näppäintä täytyy pitää kokoajan pohjassa. Tai jos hyödynnetään eleitä, niin millaista elettä tulisi käyttää?

Google Search by Voicessa puhehaut päätettiin laukaista yksittäisellä napinpainalluksella, jonka jälkeen käyttäjä lausuu hakunsa. Puheen päättyminen tunnistetaan automaattisesti hiljaisuuden perusteella. Tämän lisäksi BlackBerry ja S60 -puhelimilla on mahdollisuus määrittää manuaalisesti puheen alku ja loppu pitämällä nappia pohjassa puheen ajan. Yleinen ongelma tässä on kuitenkin tapa lopettaa painaminen ennen puheen loppua.

Seuraava ongelma oli päättää nappien sijainnista ja koosta. Käyttöliittymä on hieman erilainen eri laitteiden välillä, mutta laitteesta riippumatta mikrofoninappi on hakulaatikon oikealla puolella (kuva 5.1). Itse hakulaatikko sijaitsee yleensä ruudun ylälaidassa. Koska mikrofoninappi on melko pieni, huoleksi muodostui huomaavatko käyttäjät puheenkäyttömahdollisuutta laisinkaan. GMA:ssa päätettiinkin laajentaa napilla olevaa kosketuspinta-alaa muokkaamatta kuitenkaan napin ulkonaista kokoa. Android-puhelimissa on myös fyysinen hakunappi. Hakunappia painamalla avautuu Google Search by Voice -hakulaatikko riippumatta siitä, missä kontekstissa käyttäjä kulloinkin on. Pitämällä hakunappia pohjassa lyhyen hetken aktivoituu puhehakutoiminto.

GMA:n iPhone-versiossa on myös toteutettu mahdollisuus elelähtöiseen puhehaun aloitukseen. Eleenä toimii puhelimen nostaminen korvalle. Tilastojen mukaan monet käyttäjät pitävät ominaisuudesta, sillä yksi kolmasosa kaikista hauista aloitetaan tällä tavalla.

5.2 Palaute

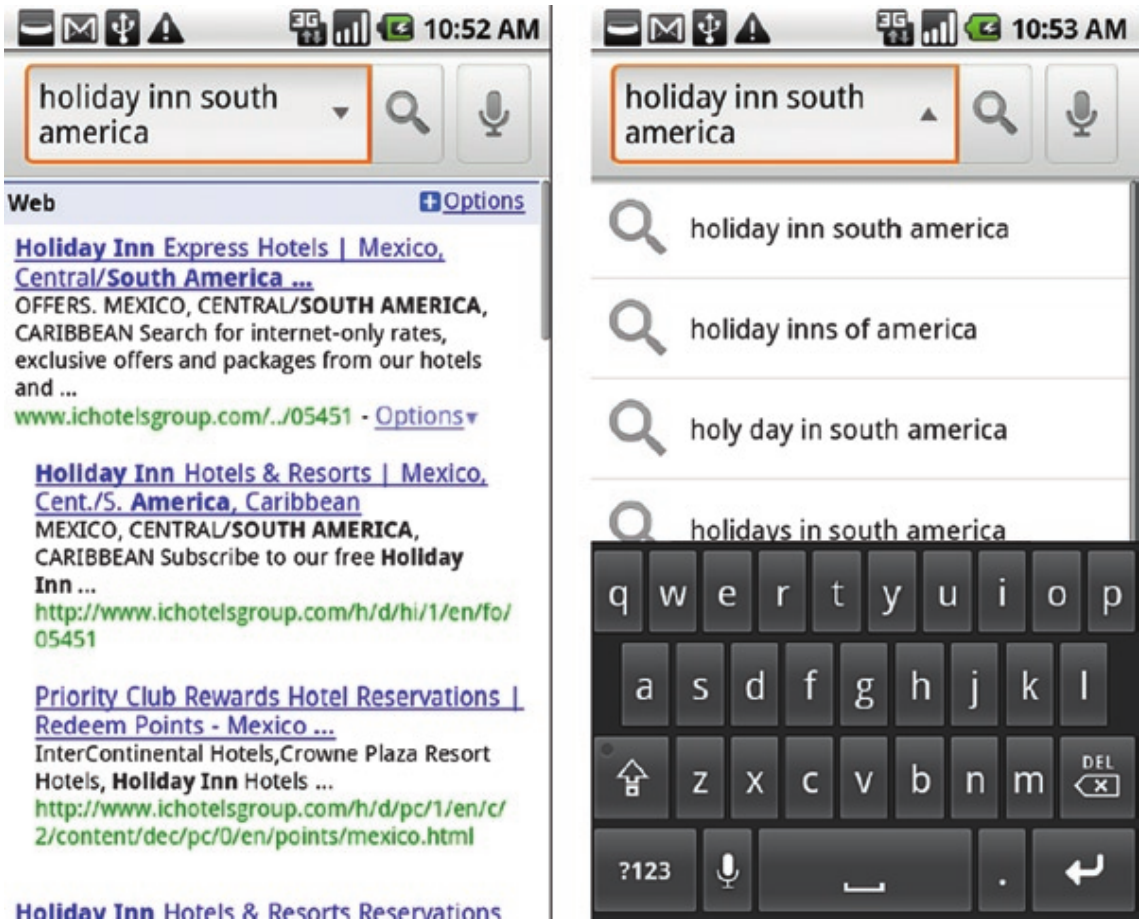
Vaikka puhehaun aloittaminen olisi käyttäjälle selkeää, tätä seuraava palaute on myös tärkeää. Eräässä testiversiossa näytettiin teksti “listening” mikrofoninapin painalluksen jälkeen osoittamaan, että käyttäjä voi nyt lausua hakukyselynsä. Sanavalinta ei kuitenkaan ollut selkeä, sillä useissa tapauksissa käyttäjä ei ymmärtänyt viestiä. Tämän seurauksena siirryttiin suoraviivaisempaan ratkaisuun käskemällä käyttäjää sanoin “Speak now”. Tästä tuli myös julkaisuversiossa käytetty sanavalinta.

5.3 Virheenkorjaus

Tunnistusvirheiden korjauksessa turvaudutaan N-Best -listaan (katso luku 3.6.2). Virheenkorjausmekanismin tarjoaminen säästää käyttäjän lauseen uudelleenpuhumiselta ja auttaa luomaan sovelluksesta positiivisemman kuvan. Listan saa näkyviin alasvetolaatikkona koskettamalla mitä tahansa kohtaa tekstikentässä (kuva 5.1). Ratkaisu ei ole kuitenkaan osoittautunut kovin tehokkaaksi, sillä Googlen tutkimuksen mukaan vain pieni osa käyttäjästä turvautuu N-Best -listaan. Tiettyä syytä tä-

hän ei osata sanoa, mutta on spekuloitu sen johtuvan esimerkiksi siitä, ettei alasvetolaatikon tarkoitus ole selvä.

Mahdollisia ratkaisuja ongelmaan on ideoitu useita. Popup-vihjeen näyttäminen tai listan välkkyminen tulosten latautuessa ovat ainakin potentiaalisia ratkaisuja. Luottamuspisteiden hyödyntäminen siten, että alhaisen luottamuksen tilanteessa listaa tuodaan voimakkaasti esille, on lisäksi yksi mahdollinen keino. Listan esilletuonnissa tulee kuitenkin tavasta riippumatta olla varovainen, sillä käyttäjän ylimääräinen häiritseminen ei ole suotavaa. Tämä pätee varsinkin tapauksessa, jossa tunnistusvirhettä ei ole tapahtunut.



Kuva 5.1: N-Best -lista Androidissa. Huomaa myös hakulaatikon ja mikrofoninapin sijainti. (Schalkwyk et al. 2010).

5.4 Pikakomennot

Samaa puhehaun käyttöliittymää käyttäen on myös mahdollista antaa joitakin google-hakuihin liittymättömiä pikakomentoja. Näitä ovat esimerkiksi “Navigate to <location>” sekä “call <contact>”, jotka mahdollistavat nopean pääsyn puhelimen eri toimintoihin. Pikakomentojen ongelmaksi koettiin kuitenkin se, että niihin liitetyt

avainsanat on helppo unohtaa varsinkin silloin, kun kyseessä ei ole tuttu toiminto. Käyttäjillä on tällöin usein taipumus ilmaista asia omin sanoin synonyymejä käyttäen (vertaa esimerkiksi “Take me to <location>”).

Ongelman ratkaisua lähestyttiin kahdella tapaa. Ensinnäkin tarjoamalla erilaisia kontekstiin perustuvia visuaalisia vihjeitä voidaan auttaa käyttäjää muistamaan oikeat avainsanat. Tämä on Googlen lyhytaikainen ratkaisua ongelmaan. Parempi ratkaisu siihen olisi kuitenkin tulkita käyttäjän tarkoitus oikein jokaisessa tilanteessa puheen semantiikkaa ja kontekstia hyödyntäen. Ideaalisesti käyttäjä voisi puhua mitä tahansa ja sovellus osaisi päätellä sen tarkoituksen. Tätä voidaan pitää pitemmän aikavälin tavoitteena, mutta sen toteuttaminen on hyvin vaikeaa. Kieli on hyvin rikasta ja sama asia voidaan ilmaista monin eri tavoin. Asia vaatisi hyvin monimutkaista semanttista käsittelyä. Vaikka tällainen järjestelmä voitaisiinkin rakentaa, se väistämättä sisältäisi ylimääräisen vaiheen, jossa puheen tulkittu tarkoitus varmistetaan käyttäjältä ennen sen toteuttamista (valitsemalla listasta). Tämä tietenkään ei ole toivottavaa. Mielenkiintoinen huomio on kuitenkin analogia normaaliin Google-hakuun. Käyttäjä sanoo haluamansa asian ja hänelle esitetään lista vaihtoehtoja, joista valita. Usein etsitty asia on vielä listan ensimmäisenä. Tästä perspektiivistä katsoen on mahdollista, että käyttäjät hyväksyisivät ylimääräisen varmistusvaiheen.

6. CASE: TEKSTIEDITORIN PUHEOHJAUS

Sublime Text 2 -tekstieditorille[1] toteutettiin sitä ohjaava puhekomentomoduuli Sublime Speech. Moduuli lisättiin Sublimen pakettienhallintaan (Package Control), joten se on asennettavissa myös suoraan tekstieditorin kautta, ja se toimii vain englanniksi.

Tarkoituksena oli tutkia tapausesimerkillä miten tällainen multimodaalinen GUI:n ja VUI:n yhdistelmä tulisi käytännössä toteuttaa ja miten modaliteetit voivat tukea toisiaan. Tämän onnistumisen merkkinä voidaan pitää tilannetta, jossa puhemoduulia käytetään mielellään, koska se parantaa tekstieditorin käyttökokemusta. Tavoitteena moduulin toteutuksessa oli myös tutkia, mihin tarkoituksiin käytettäväksi puhe soveltuu tekstieditorin yhteydessä. Toteutetusta toiminnallisuudesta tulisi siten analysoida, missä yhteyksissä puhe toimi hyvin ja missä taas GUI:n käyttö oli sujuvampaa.

6.1 Suunnitteluvaatimukset ja -tavoitteet

Perimmäisenä tavoitteena oli toteuttaa tekstieditorin graafista käyttöliittymää tukeva puhekäyttöliittymä, jonka tarkoitus ei ole korvata GUI:ta vaan tukea sitä soveltuvin osin tarjoamalla vaihtoehtoisia vuorovaikutustapoja. Puheen tarkoituksena on toimia modaliteettina, jota käytetään sen miellyttävyyden takia - ei tehokkuuden. Visiona oli luoda multimodaalinen käyttöliittymä, jossa kumpaakin modaliteettia käytetään rinnakkain ja myös toisiaan tukevalla tavalla. Vaikka käytön tehokkuus ei ollutkaan puhemoduulin varsinainen päätavoite, on se kuitenkin tärkeässä osassa puheohjauksen miellyttävyyden maksimoimiseksi. Tästä syystä oli tärkeää pyrkiä myös maksimoimaan tunnistustarkkuus sekä suunnittelemaan puhekomennot tehokasta käyttöä silmällä pitäen.

Miellyttävä käyttökokemus vaatii vuorovaikutteisten järjestelmien käytettävyysvaatimusten huomiointia (katso luku 4.2). Tästä syystä suunnittelussa kiinnitettiin huomiota myös käyttöliittymän johdonmukaisuuteen, palautteeseen, sanaston valintaan, muistikuorman minimointiin, ohjeeseen ja virheidenhallintaan. Multimodaalisuuden erikoisuutena tärkeää oli myös vastuiden oikeaoppinen jako eri modaliteettien välillä. Tätä peilaten esimerkiksi puheulostuloa päätettiin olla käyttämättä

[1] `http://www.sublimetext.com/`

mihinkään tarkoitukseen, sillä GUI:n hyödyntäminen tähän tarkoitukseen on pääasiallisesti tehokkaampaa.

6.2 Työkalut

Työkaluja puhesovellusten luomiseen on tarjolla rajoitetusti. Työpöytäsovelluksiin tarjolla olevia puheentunnistimia ovat pääasiassa CMU Sphinx, Julius, Microsoft Speech API sekä Dragon NaturallySpeaking. Valmiiden puheentunnistusjärjestelmien vähyyden lisäksi vapaasti saatavilla olevia akustisia malleja ja kielimalleja on hyvin vähän. Mallien luominen sovelluskohtaiseen käyttöön on tarkkuuden kannalta kannattavaa, mutta aiheuttaa myös lisäkustannuksia ja -työtä ja vaikeuttaa avointen puheentunnistussovellusten kehittämistä. Voxforge[2] on projekti, joka on luotu paikkaamaan vapaasti saatavien mallien puutetta. Se tarjoaa akustisia malleja ja puhekorpuksia avoimen lähdekoodin puheentunnistusmoottoreille. Voxforge toimii vapaaehtoisperiaatteella keräten audiodataa käyttäjien nauhoituksista.

6.2.1 Käytettävissä olevat puheentunnistusjärjestelmät

CMU Sphinx on Carnegie Mellon -yliopistossa kehitetty joukko avoimen lähdekoodin puheentunnistusohjelmistoja. Se sisältää Sphinx 4 ja PocketSphinx -puheentunnistusmoottorit sekä apuohjelmistoja tunnistuksessa tarvittavien mallien luontiin. CMU Sphinxin mukana tulee joitakin valmiita malleja ja myös Voxforgen avoimet mallit ovat yhteensopivia sen kanssa. Se on täysin kustomoitavissa sen käyttämien mallien suhteen ja tarjoaa näin kehittäjille vapaat kädet. Sphinx on kokonaisvaltainen ratkaisu puheentunnistustarpeisiin tarjoten ohjelmointirajapinnan puheentunnistukseen sekä työkalut omien mallien luontiin. Sphinx on tarkoitettu pääasiassa tutkimuskäyttöön ja ei täten tarjoakaan sovelluskehittäjän työtä helpottavia ominaisuuksia. Se on matalan tason järjestelmä, jonka tarkoituksena on vain kääntää puhetta tekstiksi.

Toinen avoimen lähdekoodin puheentunnistusjärjestelmä on japanilainen Julius. Sen mukana tulee vain japanilaisia malleja, joten muunkielisten käyttäjien on tehtävä omansa tai käytettävä Voxforgen malleja. Se on CMU Sphinxin tapaan hyvin muokattavissa käytettyjen algoritmien ja mallien suhteen.

Suljetun lähdekoodin ohjelmistoja ovat Microsoftin puheentunnistusrajapinta (Microsoft SAPI) sekä kaupallinen Dragon NaturallySpeaking. Toisin kuin avoimen lähdekoodin vastikkeensa ne on suunnattu pääasiassa loppukäyttäjille eikä tutkimukseen. Suljetun lähdekohdin ohjelmistoina ne eivät ole yhtä muokattavissa, mutta tarjoavat valmiin ratkaisun ilman, että kehittäjän täytyy määritellä kaikki yksityiskohdat käytettyjä akustisia malleja myöten. Ne tarjoavat avoimen lähdekoodin

[2] `http://voxforge.org`

vastineitaan enemmän ominaisuuksia puhesovellusten luomista varten ja ovat myös tunnistustarkkuudeltaan keskimäärin parempia.

6.2.2 Työkalujen valinta

Avoimen lähdekoodin puheentunnistimet CMU Sphinx ja Julius suljettiin pois työkaluvalikoimasta sen takia, että niiden kanssa työskentely jättää ohjelmoijan vastuulle hyvin paljon ja täten toteutuksesta tulisi pitkä ja monimutkainen. Lisäksi tunnistuksessa tarvittavia malleja on vähän saatavilla ja epäsopivien mallien käyttö heikentää tunnistustarkkuutta merkittävästi. Dragon NaturallySpeaking puolestaan on kaupallinen ohjelmisto, joten sitä ei haluttu hyödyntää tässä moduulissa. Valituksi puheentunnistimeksi tuli siten Microsoft Speech API, joka tosin haittapuolena rajasi moduulin vain Windows-käyttöjärjestelmälle.

Sublime Text -editori tarjoaa lisäosille ohjelmointirajapinnan python-kielellä. Tästä syystä olisi hyödyllistä käyttää kieleen helposti integroituvaa puheentunnistinta. Microsoft Speech API ei suoraan tarjoa python-rajapintaa, tosin sitä voidaan käyttää pythonin kautta Component Object Model (COM) -teknologian avulla. Dragonfly on puheentunnistukseen kehitetty python-kirjasto, joka toimii kääreenä joko Microsoft SAPI:n tai Dragon NaturallySpeakingin päällä käyttäen COM-rajapintaa. Se otettiinkin käyttöön moduulin päätyökaluna käyttäen allaolevana puheentunnistimena Microsoft SAPI:a. Dragonfly on korkean tason rajapinta puhesovelluksille tarjoten paljon työtä helpottavia ominaisuuksia. Sen valinta moduulin kehitykseen olikin hyvin luontevaa ja oli paljon kehitystyötä helpottava kirjasto. Dragonfly-kirjastoa jouduttiin itsessään hieman muokkaamaan työn edetessä, mutta tehdyt muutokset olivat hyvin minimaalisia.

6.3 Toteutettu toiminnallisuus

Puheelle voidaan havaita kolme eri käyttötarkoitusta tekstieditorin yhteydessä: tekstin sanelu, ohjelmakoodin sanelu sekä editorikohtaisten komentojen anto. Näistä mukaan otettiin tekstin sanelu sekä komentojen anto.

Ohjelmakoodin sanelu sopisi hyvin tekstieditorin ominaisuudeksi, koska suurin käyttäjäkunta lienee ohjelmoijat. Ohjelmakoodin saneluun liittyy kuitenkin monia ongelmia, jotka pohjautuvat perimmiltään siihen ettei ohjelmointikieliä ole tarkoitettu puhuttaviksi. Komentorakenteet ja kielen syntaksi eivät suoraan käänny puhutuksi kieleksi. Symbolit (muuttujat, funktiot) ovat usein sillä tavalla nimettyjä, ettei niitä pysty suoraan lausumaan. Hankaluuksia aiheuttaa myös se, ettei ohjelmointi ole luonteeltaan lineaarista. Ohjelmoinnissa esiintyy paljon tekstissä navigointia, jossa hypitään edestakaisin eri ohjelman osien välillä. Joitakin yrityksiä puheella oh-

jelmointiin on yritetty kehittää[3], mutta se pysyy silti suurena haasteena eikä ole kovin yleistä. Puheohjelmoinnin suurien haasteiden vuoksi sitä ei toteutettu tässä puhekomentomoduulissa.

Puhemoduulin tarkoituksena on tarjota helposti muistettavia äänikomentoja Sublimen ydintoimintoihin. Moduuli kytkeytyy Sublimen sovelluskohtaiseen logiikkaan kuten eri käyttöliittymäelementtien näyttämiseen/piilottamiseen. Puhekomentojen tarkoituksena on keventää käyttäjän muistitaakkaa. Sublimessa kaikki toiminnot ovat saavutettavissa näppäinlyhenteiden kautta, mutta näiden valtavan määrän takia niitä kaikkia on vaikea muistaa. Komennot löytyvät luonnollisesti myös valikkojen alta, mutta niiden selaaminen on hidasta. Sublime Speech tarjoaa tehokkaamman käyttöliittymän varsinkin valikkoja usein selaaville ja uusille käyttäjille.

Puhekomentamisen lisäksi on mahdollisuus sanella tekstiä suoraan puheella. Saneleminen kytkeytyy hyvin myös Sublimen erityissominaisuuksiin kuten mahdollisuuteen editoida samaan aikaan useampaa kohtaa tiedostossa. Tekstiä voidaan tällöin sanella usealle riville yhtäaikaisesti.

6.4 Tekninen toteutus

Puheentunnistuslisäosa toteutettiin siis python-kielellä. Moduulin toiminnallisuus lisättiin tekstieditoriin sijoittamalla sen tiedostot lisäosille tarkoitettuun kansioon, sillä Sublime Text lataa automaattisesti kaikki kyseisessä kansiossa olevat .py-päätteiset tiedostot ja ajaa niissä olevan ohjelmakoodin.

Dragonfly piilottaa paljon teknisiä yksityiskohtia alleen ja tarjoaa helppokäyttöisen rajapinnan puhekomentojen määrittämiseen. Komennot ja niihin liitetyt toiminnot määritellään sääntöluokkina, jotka on periytetty *Rule*-luokasta tai jostain sen alaluokasta kuten *MappingRule* (kuva 6.1). Säännöt lisätään yhteen tai useampaan kielioppiin (*Grammar*-luokka), jotka hoitavat kaiken kommunikoinnin puheentunnistimen kanssa.

[3] `http://sourceforge.net/projects/voicecode/`

```
class CommandRule(MappingRule):
    mapping = {
        "reset font [size]": SublimeApplicationCommand("reset_font_size"),
        "next view":         SublimeWindowCommand("next_view"),
        "[go to] line <n>":  SublimeTextCommand("goto_line"),
    }
    extras = [
        Integer("n", 1, 100000),
    ]

context = AppContext(executable="sublime_text")
grammar = Grammar("Sublime Text 2", context=context)
grammar.add_rule(CommandRule())
grammar.load()
```

Ohjelma 6.1: Sanaston luominen ja lisääminen puheentunnistimeen

Dragonfly osaa automaattisesti käyttää Microsoft SAPI:a eikä sitä tarvitse erikseen määritellä. Moduulin puheentunnistuskonteksti rajattiin vain Sublime Text -sovellukseen, joka tarkoittaa että tekstieditorin ikkunan tulee olla avoinna etualalla, jotta sille voidaan antaa puhekomentoja.

Komentojen syntaksin määrittämiseen merkkijonoina on oma merkintäkielensä, joka muistuttaa paljon Java Speech Grammar Format -kieltä. Ei-pakolliset osat merkitään hakasulkeiden (“[size]”) sisään, muuttujat tulevat pienempi ja suurempi kuin -merkkien väliin (“<n>”) ja vaihtoehtoisia ilmaisuja erotetaan pystyviivalla “|”.

Komentoihin liitetyt toiminnot ajetaan nimettyinä komentoina, jotka voivat olla Sublimeen sisäänrakennettuja tai lisäosien määrittämiä. Esimerkiksi kuvan 6.1 esittämässä tapauksessa nimettyjä komentoja ovat “reset_font_size”, “next_view” ja “goto_line”. Viimeimmän tapauksessa komennolle annetaan n-parametrin avulla rivi, jolle siirtyä.

6.5 Suunnitteluratkaisut

Suunnittelussa pyrittiin hyviin puhekäyttöliittymän periaatteisiin. Suunnittelua rajoitti tekstieditorin tarjoama lisäosarajapinta, joka oli tarpeeksi laaja moduulin toteuttamiseen mutta oli joissakin tapauksissa liian rajoittunut sekä vajavaisesti dokumentoitu. Esimerkiksi paremmalla kontekstitiedolla puheohjauksesta olisi mahdollisesti pystynyt tekemään tehokkaamman.

Eräs yleisen tason ongelma oli päättää käyttääkö jaettua (shared) vai yksityistä (in-proc) puheentunnistajaa. Jaettu tunnistin on nimensä mukaisesti jaettu kaikkien työpöytäsovellusten kesken ja sillä voi siis kontrolloida koko käyttöjärjestelmää. Sitä suoritetaan erillisessä prosessissa ja se avaa tunnetun Windows Speech Recog-

nition -sovelluksen ja sen käyttöliittymän. Yksityinen tunnistaja puolestaan on sidottu tiettyyn prosessiin ja on täysin sen hallinnassa. Microsoft suosittelee käyttämään jaettua tunnistinta, mutta Sublime Speech -moduulissa päädyttiin kuitenkin käyttämään yksityistä tunnistinta. Tämä johtuu siitä, että jaetun tunnistimen mukana tulevat myös sen oletustoiminnot ikkunoiden hallintaan ja tekstinmuokkaukseen. Näitä pidettiin haitallisena moduulin kannalta, sillä sen haluttiin olevan täysin omassa hallinnassa. Lisäksi jaetun tunnistimen mukana avautuvaa käyttöliittymää pidettiin häiritsevänä. Jaetun tunnistimen käyttö olisi myös vaatinut Windowsin asettamista käyttämään englanninkielistä käyttöliittymäkieltä, joka on iso vaatimus muunkielistä Windowsin versiota käyttäville.

6.5.1 Sanaston valinta

Komennoissa pyrittiin luonnollisen kielen ja lyhyen kielen yhdistelmään. Komennoista pyrittiin ensisijaisesti tekemään mahdollisimman ytimekkäitä vaatimalla vain välttämättömät avainsanat. Tätä rikastettiin vielä tarjoamalla sopivissa tilanteissa vaihtoehtoinen syntaksi, jossa sama asia voitiin ilmaista hieman luonnollisemmalla kielellä. Esimerkiksi halutessa siirtyä riville sata voidaan sanoa lyhyesti vain “line 100” tai luonnollisemmin “go to line 100”. Tarpeen mukaan annettiin myös mahdollisuus käyttää synonyymejä kuten esimerkiksi “decrease font size” tai “smaller font size”. Tarkoituksena on tehdä komennoista niin minimaalisia ja luonnollisia, että ne kykenee oppimaan nopeasti ja niiden unohtaminen olisi vaikeaa.

6.5.2 Komennonannon tehokkuus

Sanelumoodi sisältää komentojen toistomekanismin, jolla lausuttu komento tai komentoketju on mahdollista toistaa halutun monta kertaa. Esimerkiksi “this is text new line repeat ten times” kirjottaisi kymmenen kertaa “this is text” kukin omalle rivilleen. Yksinkertaisella toisto-operaatiolla sanelua voidaan tehostaa eikä käyttäjän tarvitse toistaa komentoja moneen kertaan.

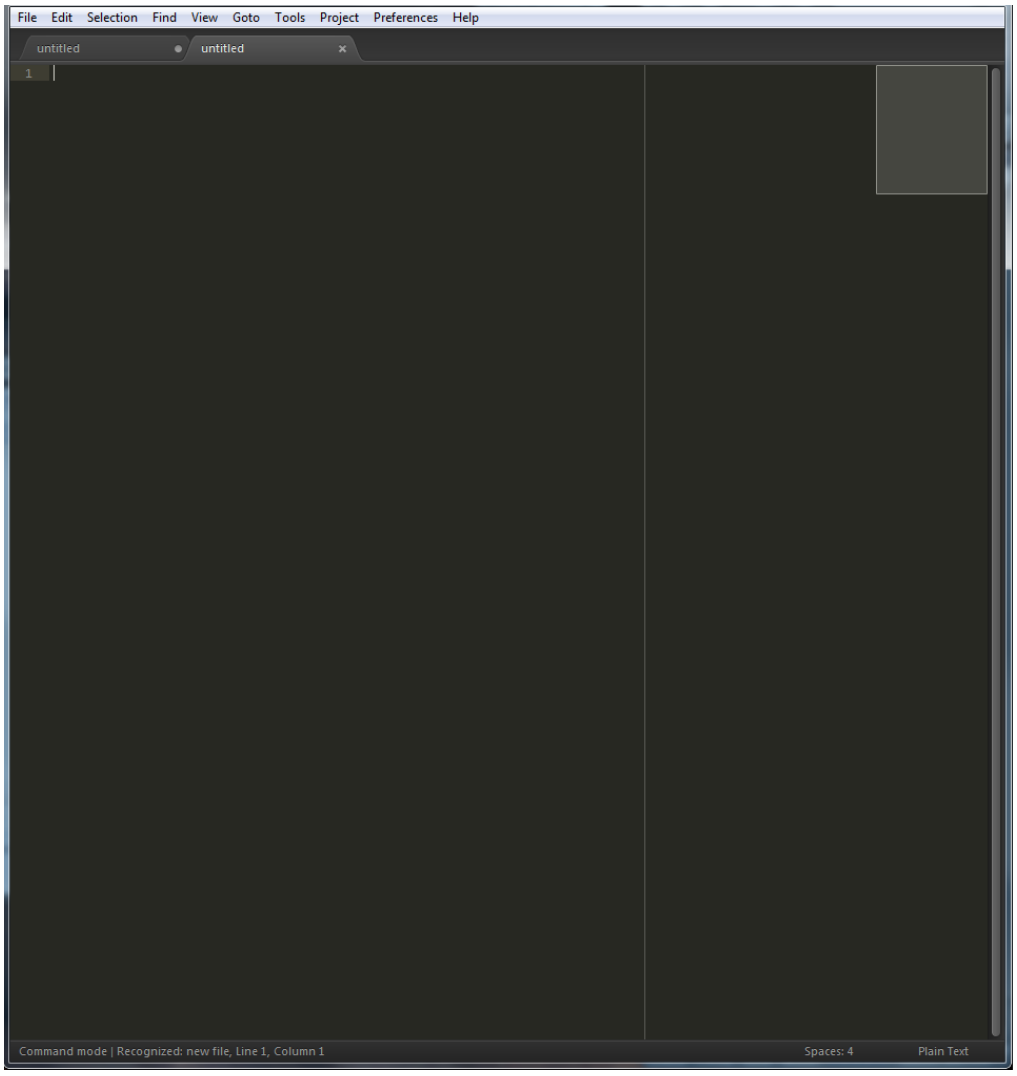
6.5.3 Tilanvaihdot

Koska Sublime Speech ei ole tarkoitettu editorin pääasialliseen hallintaan, on käyttäjällä hyvä olla mahdollisuus poistaa se hetkellisesti käytöstä. Tämä estää ylimääräistä melua laukaisemasta komentoja, jotka haittaavat käyttäjän työtä. Puheentunnistin onkin mahdollista sammuttaa kokonaan tai pistää nukkumaan, jolloin se on vielä päällä, mutta ei reagoi muihin komentoihin kuin sammuttamiseen tai herättämiseen. Puheentunnistus toimii siis avoin mikrofoni -periaatteella, jota on varioitu tarjoamalla nukkumis- ja pois päältä -tilat. Nämä lisätilat ovat eräänlainen klikkaa

puhuaksesi -moodin muoto. Avoin mikki -moodin valinta pääasialliseksi toimintaperiaatteeksi perustuu multimodaalisen synergian maksimointiin. Muissa moodeissa aikaa ja vaivaa kuluu modaliteetin vaihtoon, mikä heikentää työskentelyn tehokkuutta.

6.5.4 Tila ja palaute

Käyttäjän tulisi aina tietää missä tilassa järjestelmä on. Tätä varten Sublime Speech näyttää puheentunnistuksen tilan alhaalla tilarivillä. Puheentunnistimen moodi (komento tai sanelu) tai sen tila (pois päältä tai nukkumassa) lukee rivillä selväkielisesti. Tilariville kirjoitetaan myös lyhyeksi hetkeksi tunnistetut ja tunnistamattomat komennot palautteenantoa varten. Näin käyttäjä pystyy seuraamaan järjestelmän tilaa ja saa palautetta komentoihinsa.

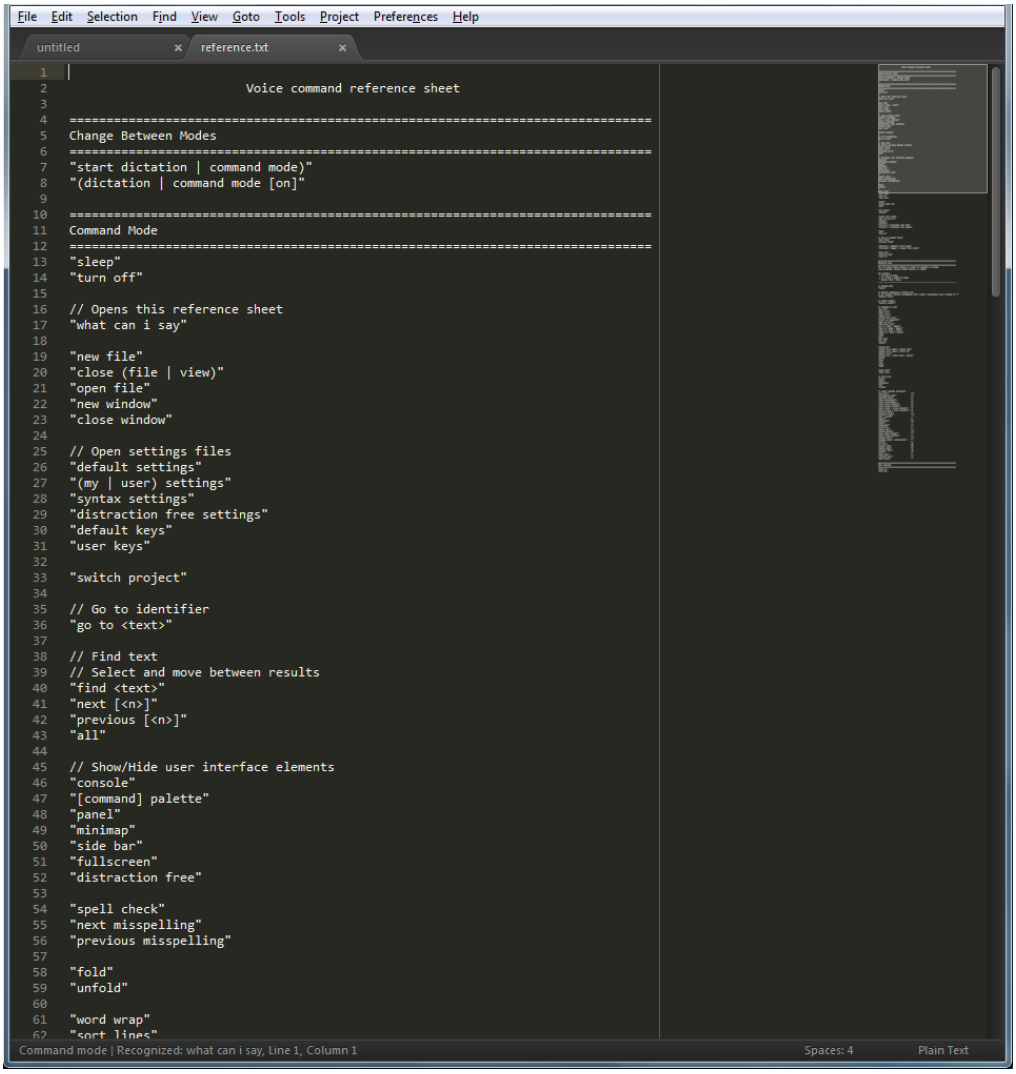


Kuva 6.2: Sublime Text 2 -tekstieditori. Alhaalla tilarivillä näkyy puheentunnistuksen tila (moodi) sekä äskettäin tunnistettu komento.

Tunnistetusta komennosta näytetään viesti “Recognized: <komento>” ja Sublime Text lisäksi joissain tilanteissa näyttää automaattisesti tilarivillä mikä toiminto suoritettiin. Tunnistamattomasta komennosta näytetään “What was that?” samalla tapaa kuin Windows Speech Recognition -ohjelma, joten viestin pitäisi olla selvä. Äänipalautetta ei käytetä ollenkaan, koska jotkut käyttäjät kokevat sen häiritseväksi ja lisäksi se voi joissakin tilanteissa olla yleisesti epätoivottua.

6.5.5 Läpinäkyvyys

Tärkeä ratkaisu moduulin läpinäkyvyyden lisäämiseen oli antaa tietoa käytettävissä olevista komennoista. Tämä toteutettiin kirjoittamalla tekstitiedostoon kaikki käytettävissä olevat komennot ja selittämällä ne lyhyesti. Käyttäjä saa referenssin näkyviinsä joko graafisen käyttöliittymän kautta tai puhekomennolla “What can I say?”. Referenssi (kuva 6.3) näytetään tekstieditorissa uudella välilehdellä.



Kuva 6.3: Komentoreferenssi. Useimmat komennot ovat tekstieditorin käyttäjille itsestään selviä, mutta joitakin on myös lyhyesti selvennetty.

6.6 Toteutus

Moduulia suunniteltaessa haluttiin lähtökohtaisesti tarjota mahdollisimman korkea tunnistustarkkuus. Tämä huomioitiin käyttämällä kontekstisidonnaisia sanastoja, jolloin kulloisessakin tilanteessa tunnistin ymmärsi vain komentoja, jotka olivat tilanteeseen sopivia. Tämän lisäksi tärkeätä oli myös minimoida komentojen väliset konfliktit ääntämyksessä, joten komennoista pyrittiin tekemään erikuuloisia, ettei tunnistin sekoittaisi niitä keskenään.

Moduulin prototyyppiä kokeiltaessa havaittiin, että tunnistin saattoi häiriintyä taustamelusta ja suorittaa satunnaisia komentoja. Yleensä nämä olivat harmittomia, mutta häiritseväksi koettiin, jos tunnistin tässä tilanteessa saneli satunnaista tekstiä. Käyttäjä usein tässä tilanteessa käytti puhetta vain yksinkertaisten komentojen antoon ja kirjoitti kaiken tekstinsä käsin.

Tämän korjaamiseksi jaettiin moduulin toiminnallisuus kahteen moodiin: komentojen antoon ja saneluun. Komentomoodissa sallitaan vain editorin ohjaamiseen käytetyt komennot eikä sallita mitään komentoa, joka muokkaisia tekstiä millään tavalla. Sanelumoodissa puolestaan komentojen anto kiellettiin ja kaikki puhekomennot liittyvät tekstin saneluun ja muokkaamiseen sekä siinä navigointiin. Moodien erottamisesta on myös hyötyä tekstin sanelussa, sillä siten saatiin rajattua osa sanelua häiritsevistä komennoista pois. Käyttäjän halutessa sanella jonkin lausahduksen, joka on samalla myös äänikomento, ymmärrettiin lausahdus komennoksi eikä saneltavaksi tekstiksi. Toiminnallisuuden jakoa puoltaa myös se, että osa käyttäjistä haluaa käyttää moduulia vain komentojen antoon. Tällöin saneluominaisuutta ei käytetä, ja sen sisältyminen moduulin yleiseen toimintaan häiritsee käyttäjää.

6.6.1 Komentomoodi

Komentomoodiin pääsee lausumalla “start command mode” tai “command mode on”. Moodissa sallittu komentoja ovat vain yleiset editorin ohjausta tukevat komennot. Tunnistustarkkuuden maksimoimiseksi mahdolliset vapaat tekstikentät on pyritty minimoimaan. Ainoa komento, joka hyväksyy vapaata kieltä on “find <text>”. Komento etsii aktiivisesta dokumentista haluttua tekstiä.

Komentomoodissa voidaan näyttää tai piilottaa eri käyttöliittymäelementtejä niiden nimiä vastaavilla komennoilla kuten “console”, “[command] palette”, “panel”, “minimap” ja “side bar”. Komennot on pidetty lyhyinä eikä niihin kuulu eteen mitään avainsanoja kuten “show” tai “hide”. Muita yleisiä komentoja ovat “new file | window”, “close file | window” ja “‘save”. Moodiin sisältyy myös monia muita yleensä yhden tai kahden sanan komentoja. Komentojen intuitiivisuuden ja lyhyyden vuoksi ne ovat helposti muistettavia toisin kuin niihin liitetyt näppäinyhdistelmät, jotka voivat olla mitä tahansa.

6.6.2 Sanelumoodi

Sanelumoodiin pääsee lausumalla “start dictation mode” tai “dictation mode on”. Tekstin sanelu on tehty yksinkertaiseksi: sitä ei laukaise mikään komento vaan kaikki muuhun sopimaton lausahdus tulkitaan saneluksi. Saneluun liittyvät erikoisuudet on myös otettu huomioon. Numeroiden saneleminen tapahtuu avainsanalla “numeral”, jonka jälkeen lausutaan haluttu numero. Tällöin numero kirjoitetaan numeromuodossaan eikä sanallisessa muodossa, joka tehtäisiin jos avainsanaa ei käytetä.

Erikoismerkit on huomioita samaan tapaan kuten numerotkin. Niitä ei tosin tarvitse aloittaa avainsanalla vaan sanomalla yksinkertaisesti merkin nimen. Moduulissa on toteutettu yleisimmät merkit kuten “exclamation mark”, “comma”, “open parenthesis” ja “close parenthesis”. Näppäinpainallusten emulointi toteutuu identtisesti erikoismerkkien tapaan. Tuetut näppäimet ovat “enter”, “space”, “backspace”, “tab” ja “escape”.

Eri avainsanojen mukaanottaminen tuo ongelmaksi tilanteen, jossa halutaan sanella kyseinen avainsana. Tämä korjattiin lisäämällä yksi avainsana lisää. Lausumalla “literal” ja sen jälkeen mitä tahansa, tulkitaan se pelkäksi saneluksi eikä eri komennoiksi. Esimerkiksi lausumalla “literal numeral” tulostetaan vain “numeral”.

Tekstissä voidaan myös navigoida yksinkertaisilla komennoilla. “up | down | left | right” siirtää kursoria haluttuun suuntaan. Komennon perään voidaan liittää myös numeromääre (“right 5”), jolloin suuntaan siirrytään haluttu määrä. Perinteisen merkkipohjaisen liikkumiseen lisäksi voidaan siirtyä sanoittain tai sivuttain. “left | right <n> words” siirtää valitsinta halutun monta sanaa vasemmalle tai oikealle. “page up | down <n>” tai “up | down <n> pages” -komennoilla voidaan navigoida dokumenttia sivutasolla. Myös muita navigointikomentoja, kuten “home”, on olemassa.

Tekstin valitsemiseen on lyhyesti vain komennot “select all” sekä “select <n> lines up | down”. Tämä on kenties moduulin heikoin osa-alue, ja tehokkaaseen tekstin valintaan vaaditaankin käytännössä hiirtä. Komentomoodissa on lisäksi tarjolla “find”-komento, jolla tekstissä pystytään etsimään ja valitsemaan osia. Se ei kuitenkaan toimi kontekstiperustaisesti siten, että ottaisi huomioon vain aktiivisessa dokumentissa esiintyvät sanat. Tällöin sen käyttö on virhealtista eikä tulos ole haluttu ilman täydellistä ääntämystä. Kaikenkaikkiaan moduulin tekstinvalintaominaisuuksissa olisi vielä parantamisen varaa.

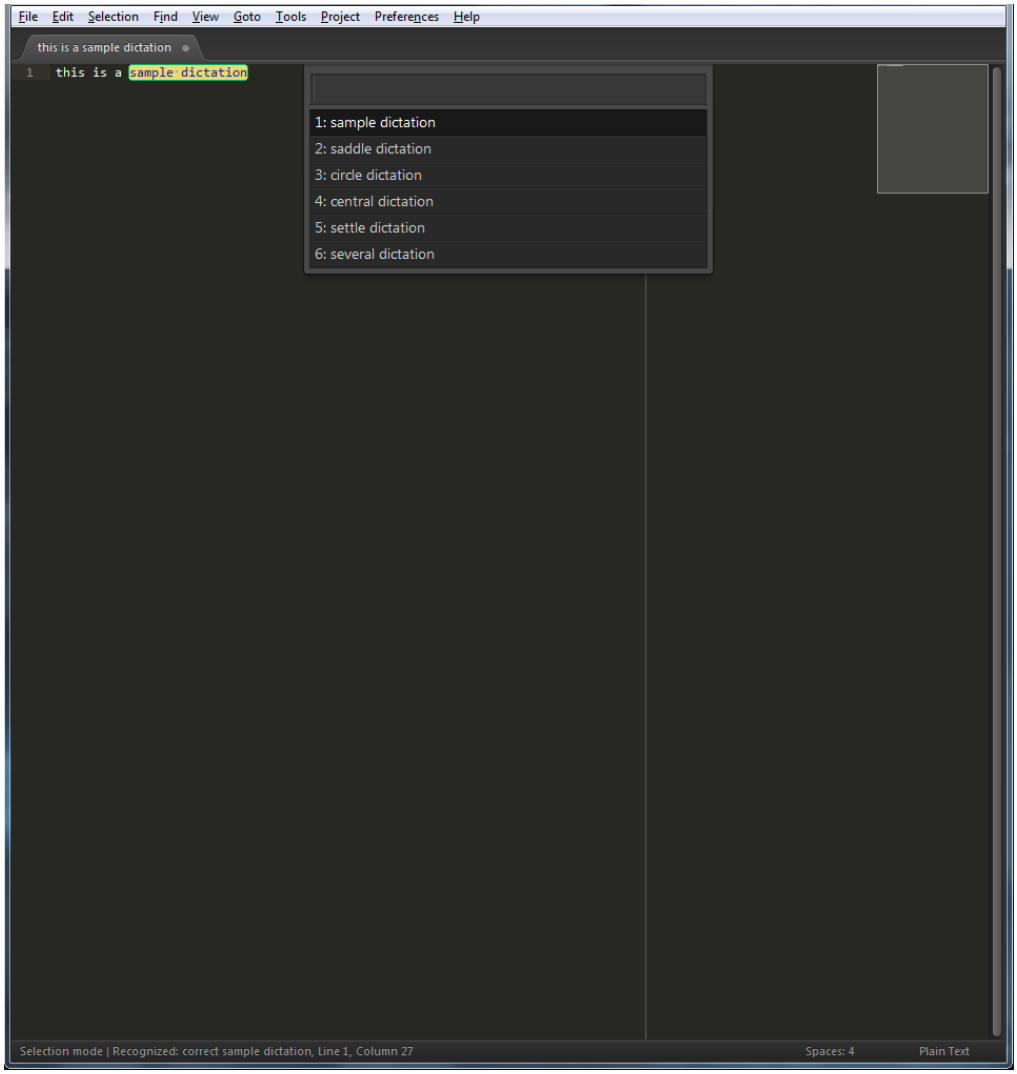
6.6.3 Virheenkorjaus

Tehokkaassa sanelussa vaaditaan jokin mekanismi, jolla korjata väistämättä syntyvät tunnistusvirheet. Sublime Speech -moduulissa toteutettiin oma puheella toimiva virheenkorjausmekanismi, joka integroituu saumattomasti Sublime-tekstieditoriin

käyttämättä omia ponnahdusikkunoita. Käyttämällä vain tekstieditorin omia käyttöliittymäelementtejä saadaan moduuli vaikuttamaan kiinteästi osalta Sublimen toimintaa.

Virheenkorjauksessa toteutettiin mahdollisuus korjata äskettäin saneltua lausetta. Tärkeä rajoitus on juurikin tämä, että suora korjaus on mahdollistettu vain viimeisimmälle lausahdukselle. Muun vanhemman tekstin korjauksessa joudutaan se ensin valitsemaan sekä poistamaan ja lopuksi sanelemaan koko teksti uudestaan.

Tekstinkorjaus aloitetaan lausumalla “correct” ja sen jälkeen mahdollisesti jokin osa saneltua lausetta. Pelkkä yksittäinen “correct” olettaa, että käyttäjä haluaa korjata koko lauseen. Tämän jälkeen moduuli korostaa tekstinosan, johon korjaus kohdistuu ja näyttää listan vaihtoehtoisia sanoja tai sanayhdistelmiä, jotka sopivat kyseiseen kohtaan. Mukaan otetaan enintään 6 vaihtoehtoa (tai 5, sillä ensimmäinen vaihtoehto on aina kyseinen tekstinosa itse).



Kuva 6.4: Virheenkorjaus.

Esiin ilmestyvästä listasta (kuva 6.4) valitaan oikea vaihtoehto, jolloin valittu tekstinosa korvataan kyseisellä vaihtoehdolla. Vaihtoehto voidaan valita hiirellä, näppäimistöllä tai äänikomennolla lausumalla oikean vaihtoehdon numero. Tilanteessa sovelletaan kontekstisidonnaista sanastoa. Esimerkiksi kuvan 6.4 tilanteessa sallitut äänikomennot ovat vain numerot 1-6. Korjaus voidaan myös perua lausumalla “cancel”.

Virheitä korjatessa on kannattavaa korjata samalla kertaa koko haluttu tekstinosa, eikä osittain esimerkiksi sana kerrallaan. Tämä johtuu siitä, että tunnistin pystyy antamaan parempia vaihtoehtoja, kun se tietää mille kontekstille vaihtoehto halutaan löytää. Korjattaessa sana kerrallaan osa kontekstista on väärä, sillä siinä esiintyy vielä korjaamattomia sanoja.

7. ARVIOINTI

Edellisessä luvussa kuvatun puhekomentomoduulin toteutuksen onnistuminen on tarkoituksenmukaista arvioida käytettävyyden näkökulmasta. Käytettävyysarvioinnin perusteella voidaan poistaa pahimpia ongelmia ja kehittää moduulia eteenpäin. Tässä luvussa on kuvattu arvioinnin metodit, tulokset, johtopäätökset ja esitetty myös joitain parannusehdotuksia.

Sublime Speech -moduulin käytettävyyttä arvioitiin yhden hengen asiantuntijaarvioinnilla, jossa moduulin puhekäyttöliittymä käytiin läpi heuristisen arvioinnin menetelmin. Tarkoituksena oli selvittää miten hyvin moduuli vastasi luvussa 4.2 esiteltyihin käytettävyysvaatimuksiin. Arviointiprosessi suunniteltiin luvussa 4.1 kuvatun arviointikehyksen pohjalta.

7.1 Arviointisuunnitelma

Lopullisessa heuristisessa arvioinnissa hyödynnettiin luvussa 4.3 kuvattuja Nielsenin heuristiikkoja. Heuristiikkojen soveltamiseksi puhekäyttöliittymien tapaukseen tutkittiin arvioinnissa myös Farinazzo et al. (2010) omassa tutkimuksessaan kehittämää puhekäyttöliittymien heuristista tarkistuslistaa.

Toteutettu puheohjausmoduuli käytiin läpi kokonaisuudessaan seuraavalla listalla. Jokaista pääotsikkoa tarkasteltiin muutaman alakysymyksen kautta:

- Järjestelmän tilan näkyvyys
  - Antaako sovellus palautetta jokaiseen käyttäjän toimintoon?
  - Kertooko järjestelmä puheentunnistuksen onnistumisesta tai epäonnistumisesta?
  - Onko puheentunnistuksen tila aina näkyvissä?
- Järjestelmän tulee vastata ulkomaailmaa
  - Onko järjestelmän käyttämä kieli helppoa ja luonnollista käyttäjän näkökulmasta?
- Käyttäjän vapaus ja hallinnan tunne
  - Pääseekö moniaskelisista toiminnoista pois kesken kaiken?

  - Voiko toiminnot perua ja toistaa?
  - Tunteeko käyttäjä hallitsevansa sovellusta?
- Johdonmukaisuus ja standardit
  - Vastaavatko puhekomennot vastaavien toimintojen nimiä kuten ne on tekstieditorissa esitetty?
  - Ovatko puhekomennot loogisia?
  - Ovatko puhekomennot yhdenmukaisia?
- Virheiden välttäminen
  - Kuulostavatko puhekomennot tarpeeksi erilaisilta?
  - Ovatko puhekomennot tarpeeksi erilaisia toiminnaltaan?
  - Käytetäänkö kontekstisidonnaista sanastoa aina, kun se on mahdollista?
  - Onko tunnistustarkkuus tarpeeksi korkea, eli tapahtuuko tunnistusvirheitä vain harvoin?
- Tunnistamista, ei muistamista
  - Ovatko käytettävissä olevat komennot aina näkyvissä kulloisessakin kontekstissa?
- Käytön tehokkuus ja joustavuus
  - Onko järjestelmän viestit mukautettu käyttäjän tason mukaan?
  - Voiko käyttäjä määrittää omia puhekomentojaan?
- Esteettisyys ja minimalistinen suunnittelu
  - Onko näkyvillä vain tarpeellinen informaatio?
  - Sulautuuko puhekäyttöliittymä saumattomasti osaksi tekstieditorin käyttöliittymää?
- Ohjeet ja dokumentointi
  - Tarjoaako sovellus täydellisen ja yksityiskohtaisen ohjeen sen käyttämiseen?
  - Onko ohje helppolukuinen?
  - Onko ohjeesta eritasoisia versioita ongelman monimutkaisuuden mukaan?

7.2 Tulokset

Arviointi suoritettiin hiljaisessa ympäristössä työpöytäkoneella. Mikrofonina toimi irrallinen web-kamera, jonka etäisyys käyttäjästä oli noin 50-100cm. Mikrofoni ei siis häirinnyt käyttäjää millään tavalla, mutta toisaalta parempi mikrofoni lähellä käyttäjän suuta olisi vähemmän häiriöaltis. Arviointitilanteen lisäksi arvioinnissa hyödynnettiin vanhaa tuntemusta järjestelmästä ja siitä aikaisemmin tehtyjä huomioita.

Järjestelmän tilan näkyvyys todettiin hyväksi. Kaikista toiminnoista seuraa jokin visuaalinen muutos käyttöliittymässä. Tätä ei tarvinnut huomioida puhemoduulin toteutuksessa, sillä tekstieditori itsessään on toteutettu sitä tukien. Tunnistetusta tai tunnistamattomasta komennosta näytetään tilarivillä vastaava ilmoitus. Tila on näkyvissä aina tilapalkissa.

Järjestelmän vastaa myös ulkomaailmaa. Komennoissa on pyritty luonnollisuuteen ja yksinkertaisuuteen.

Käyttöliittymä tukee käyttäjän vapautta. Moniaskelisista toiminnoista pääsee pois kesken kaiken. Tällaisia toimintoja oli vain yksi, sanelunkorjaus. Sen voi perua lausumalla “cancel”. Toimintoja voi myös perua ja toistaa. Tekstieditoriin on rakennettu toista/kumoa -toiminto, jonka voi laukaista myös äänikomennoilla “redo” tai “undo”. Käyttäjä tuntee hallitsevansa sovellusta. Tässä ratkaiseva päätös oli toiminnallisuuden jako komento- ja sanelumoodiin. Hallinnantunne kasvoi, kun häiriömelu ei laukaissut vahingossa tekstinsanelua.

Käyttöliittymä todettiin johdonmukaiseksi. Puhekomennot vastaavat vastaavien toimintojen nimiä ja ne ovat loogisia. Komennot ovat myös keskenään yhdenmukaisia. Esimerkiksi käyttöliittymäelementtien näyttämisen ja piilottamisen komentoja ei ole sekoitettu siten, että jokin vaatisi “show”-sanan käyttöä ja jokin toinen ei. Toiminto hoituu yksinkertaisesti lausumalla elementin nimi. Samalla tapaa muutkin komennot kuten navigointikomennot ovat johdonmukaisesti määriteltyjä.

Virheiden välttäminen on toteutettu kohtalaisen hyvin. Puhekomennot ovat tarpeeksi erilaisia, jotta tunnistin ei helposti sekoita niitä keskenään. Kontekstisidonnaista sanastoa olisi voinut käyttää enemmän, sillä se on nyt toteutettu vain osassa sille sopivissa tilanteissa. Konteksti on otettu huomioon esimerkiksi sanelunkorjauksessa, jossa valitaan oikea vaihtoehto listasta, jolloin on sallittua lausua vain halutun vaihtoehdon numero. Toisaalta esimerkiksi komentomoodin komennossa “view <n>”, joka siirtää näkymän haluttuun välilehteen (n = numero), ei käytetä kontekstia. Komennossa on mahdollista lausua n:n paikalla mikä tahansa numero eikä sitä rajata välilehtien lukumäärään. Käytännössä numero tunnistetaan useimmiten oikeaksi, sillä se on kuitenkin rajattu kokonaislukuihin eikä rajoittamattomaan sanastoon. Komentomoodin tunnistustarkkuus onkin yleisesti erittäin hyvä, mikä onkin

odotettua, sillä siinä on hyvin rajoitettu sanasto. Sanelumoodissa tunnistustarkkuus ei luonnollisesti ole samalla tasolla ja usein virheitä tapahtuukin. Sanelumoodin tunnistustarkkuuden olisi syytä olla korkeampi, mutta tämä vaatisi jo teknologiallisia edistyksiä.

Puhemoduulin aiheuttama muistikuorma havaittiin hieman raskaaksi. Kulloisessakin kontekstissa tarjolla olevia komentoja ei ole näkyvillä. Puhekomentoreferenssistä käyttäjä voi tarkistaa käytettävissä olevat komennot, mutta se vaatii aina erillistä tarkastusta. Ongelmaksi koetaan eritoten sanelunkorjauksessa oleva selkeä puute opastuksessa. Käyttäjän tulisi valita oikea vaihtoehto listasta lausumalla vaihtoehdon numero, mutta missään ei ole opastettu lausumaan numeroa. On täysin mahdollista ettei käyttäjä edes havaitse tilanteessa olevaa mahdollisuutta äänikomentoihin. Yleisestikin olisi kenties hyödyllistä esittää visuaalinen kontekstisidonnainen komentoreferenssi, joka mukautuu kulloiseenkin tilanteeseen. Tällainen referenssi voisi esimerkiksi olla aina näkyvillä erillisessä ikkunassa tai sen voisi tuoda esiin jollakin avainkomennolla. Tehokkaan komentoreferenssin puutetta hieman tasapainottaa äänikomentojen loogisuus ja helppo arvattavuus, mutta tämä ei ole tyydyttävä ratkaisu ongelmaan.

Käytön joustavuudessa olisi parannettavaa. Omien puhekomentojen määrittäminen olisi kenties hyödyllinen ominaisuus, mutta tätä ei ole suoranaisesti tuettu. Koska moduuli on avointa lähdekoodia ja toteutettu dynaamisesti tulkittavalla pythonkielellä, omia komentoja voi lisätä muokkaamalla itse lähdekoodia. Tämä on kuitenkin huono ratkaisu, koska esimerkiksi päivitettäessä moduulia uuteen versioon kaikki käyttäjän omat muokkaukset ylikirjoitetaan. Omien komentojen määrittämiseen tulisikin olla helpompi tapa. Mahdollinen ratkaisu ongelmaan olisi mahdollistaa puhekomentojen liittäminen Sublime Text -editorin makrotiedostoihin. Käyttäjä voisi tällöin kirjoittaa oman makronsa ja määrittää puhekomennon, jolla se ajetaan.

Puheohjauksen lisäämä visuaalinen käyttöliittymä on esteettinen ja minimaalinen. Se sulautuu luontevasti yhteen tekstieditorin oman käyttöliittymän kanssa, koska se käyttää tekstieditorin omia käyttöliittymäelementtejä.

Komentomoduulin ohjeissa olisi paljon parannettavaa. Jo aikaisemmin mainittu kontekstisidonnainen ohje olisi hyödyllinen. Tästä huolimatta toteutetussa staattisessa ohjeessakin on käytettävyysongelmia. Ohje on täydellinen lukuunottamatta sitä, että se ei mainitse mahdollisuutta valita sanelunkorjausvaihtoehto numeroin. Ohje ei kuitenkaan ole yksityiskohtainen tai helppolukuinen. Siinä ei kerrota jokaisen komennon kohdalta, mitä se tekee. Ohjetta ei myöskään ole helppoa lukea, sillä se on kirjoitettu yhtenä suurena listana tavallista tekstiä ilman tekstinmuokkausta tai syntaksinkorostusta.

7.3 Multimodaalisuus ja puheen käyttökohteet

Yhtenä tavoitteena oli tutkia, mihin käyttötapauksiin puhe sopii ja ei sovi tekstieditorin yhteydessä. Tätä tarkoitusta varten on hyvä arvioida puheen sopivuutta toteutettuihin toiminnallisuuksiin.

Tekstinsanelussa puheen rooli on ristiriitainen. Tunnistuksen epätarkkuus sekä osin myös toteutuksen minimaaliset saneluominaisuudet tekevät prosessista tehokäyttäjälle turhauttavan ja epävarman. Lisäksi erikoismerkkien ja lukujen lausuminen on epäkäytännöllistä, mutta toisaalta tavallisessa tekstissä näitä esiintyy vain vähän. Selville ääntäjille, hitaille kirjoittajille tai erityisryhmille kuten liikuntarajoitteisille sanelu voi kuitenkin olla tehokkaampi tai ainoa keinoa tekstintuottoon. Tekstieditorin puhesanelumahdollisuus voi kuitenkin olla harvoin käytetty ominaisuus, johon suurin osa käyttäjistä ei turvaudu. Tämä varsinkin ottaen huomioon, että tekstieditoria ei perinteisesti käytetä normaalin tekstin tuottoon vaan ohjelmointi- ja merkintäkielten kirjoittamiseen. Nämä kielet poikkeavat merkittävästi normaalista kielestä ja eivät sovellu hyvin perinteiseen tekstinsaneluun.

Sen sijaan puhe sopi paremmin lyhyiden äänikomentojen antamiseen. Ihmisten luontainen mieltymys äänen käyttöön edesauttoi käyttökokemuksen miellyttävyyttä. Usein komennon antaminen on miellyttävämpää puheella kuin näppäinlyhenteillä vaikka näppäimistön käyttö onkin hieman nopeampaa. On hyvä kuitenkin huomata että kaikki äänikomennot eivät ole kovin käytettäviä. Komennot, joita joudutaan toistamaan tavoitteen saavuttamiseksi, ovat usein hankalakäyttöisiä ja tehottomia. Tekstieditorin yhteydessä tällaisia komentoja olivat ainakin tekstissä navigoimiseen käytetyt komennot (“right 5”, “down 2”). Tällaiset toiminnot on helpompaa suorittaa osoitinlaitteella (hiirellä) käytännössä jokaisessa tapauksessa. Seurauksena tästä myös olemassaolevan tekstin muokkaus on hankalaa tekstissä navigoinnin ongelmista johtuen. Moduulissa toteutettua merkki-, sana- ja rivipohjaista navigointiratkaisua parempi vaihtoehto olisikin kenties hakuun perustuva navigointimalli, jossa käyttäjä lausuu tekstissä ilmenevän sanan tai lauseen, jonne on tarve siirtyä.

7.4 Johtopäätökset

Yleisesti ottaen komentomoduulin käytettävyys on hyvä kokeneella käyttäjälle. Suurin ongelma lienee uusien käyttäjien kohdalla, sillä moduulin puutteellinen ohje ei tue helppoa opittavuutta. Epäilyn alaisena on, että käyttäjän kokonaistehokkuus laskee puhekomentojen käyttöönoton myötä. Toisaalta käyttäjän opittua käytettävissä olevat komennot tehokkuuden oletetaan nousevan ja ylittävän vanhan tason.

Löydettyjen käytettävyysongelmien korjaaminen parantaisi moduulin opittavuutta ja muistettavuutta. Selkeän ohjeen kirjoittaminen olisi tässä suurimmalla prioriteetillä. Muut parannukset kuten käytön joustavuuden lisääminen eivät ole tässä

prosessissa yhtä tärkeitä, mutta olisivat kenties omiaan houkuttelemaan useampia käyttäjiä.

Arviointi itsessään on osin vajvainen arviontimenetelmästä johtuen. Arvioinnissa löydetyt ongelmat ovat täysin valideja, mutta suoritetun asiantuntija-arvioinnin lisäksi olisi käytettävyysarvioinnissa syytä tehdä myös käyttäjätestejä. Asiantuntijamenetelmin käytettävyysongelmia voidaan löytää vain tiettyyn pisteeseen saakka ja ottamalla käyttäjät mukaan saataisiinkin syvempää ja aidompaa tietoa asiasta. Arvioinnin kattavuuden ja luotettavuuden parantamiseksi toisena vaihtoehtona olisi myös suorittaa useampia asiantuntija-arviointeja, sillä yksinäinen arvioija ei pysty löytämään kaiken kattavaa otantaa ongelmista. Kuitenkin käyttäjän ottaminen mukaan arviointiprosessiin olisi suositeltavin vaihtoehto, sillä asiantuntija-arvioissa näkökulma on liian kapea-alainen.

7.5 Jatkokehitys

Toteutettu arviointi ei ottanut kantaa puheohjauksen kattavuuteen tai hyödyllisyyteen. Jatkotutkimus asiasta olikin siten tarpeen. Ilman tarkempaa tutkimustakin on selvää, että komentomoodissa tarjolla olevat toiminnot eivät kata kaikkia tekstieditorin ominaisuuksia. Puheohjaus on siten luonteeltaan tukeva ja toissijainen modaliteetti. Toteuttamalla laajemman otannan Sublime Text -editorin toimintoja voitaisiin parantaa puheohjauksen kattavuutta.

Sanelumoodia kyettäisiin myöskin parantamaan. Esimerkiksi tekstinvalintaominaisuuksia olisi syytä kehittää eteenpäin. Sanelu ei myöskään tue automaattista tekstinmuokkausta, siten että esimerkiksi lauseen aloittava sana olisi kirjoitettu isolla kirjaimella. Tämäntyyppinen älykäs muokkaus olisikin kenties hyödyllistä. Lisäksi sanelumoodin tukemaa näppäinpainallusten emulointia ja erikoismerkkien kirjoitusta olisi syytä laajentaa kattamaan suuremman otannan eri vaihtoehtoja.

Sanelunvirheiden korjaus oli toteutettu melko yksinkertaisesti. Käyttäjä pystyi korjaamaan vain viimeksi lausumansa virkkeen. Tämä on melko suuri rajoitus ja olisikin syytä laajentaa korjausmahdollisuutta kaikkeen lausuttuun puheeseen.

Teknisestä näkökulmasta katsoen komentomoduulia vaivaa huono siirrettävyys. Sublime Text 2 -tekstieditori itsessään on tarjolla Windows, Linux ja OS X -käyttöjärjestelmille. Komentomoduulissa puheentunnistimeksi valittu Microsoft SAPI toimii kuitenkin vain Windows-ympäristössä, joten muiden käyttöjärjestelmien käyttäjät eivät kuulu moduulin kohdeyleisöön. Käytännössä moduulin kohdeyleisöä on noin puolet kaikista tekstieditorin käyttäjistä, sillä editorin Package Control -lisäosan asennustilastojen mukaan 48%:lla käyttäjistä (1,3 miljoonaa) on Windows (wbond).

Toinen teknisen näkökulman parannus olisi mahdollistaa puheohjauksen lokalisaatio. Toteutuksessa lokalisaatiota ei ole otettu huomioon ja ainoa mahdollinen kieli onkin englanti. Sublime Text -editori ei tosin itsekään tue suoraan lokalisaa-

tiota vaan siitä on jakelussa vain englanninkielinen versio. Editorin käyttöliittymän tekstejä voidaan kuitenkin vaihtaa asetustiedostoja muokkaamalla vaikka se ei olekaan kovin käytännöllinen ratkaisu. Tällä hetkellä puheohjauksen rajaaminen vain englanniksi kenties on hyväksyttävä ratkaisu, mutta mahdollisia tulevia tarpeita varten lokalisaation mahdollistaminen olisi syytä ottaa tarkastelun alle. Monikielisyyden mahdolliseksi ongelmaksi voi tosin nousta puheentunnistuksen tason vaihtelu eri kielten välillä sekä puheentunnistimien kielituki. Esimerkiksi morfologisesti voimakkaiden kielten, kuten suomi, puheentunnistukseen tässä työssä esitetyt menetelmät eivät välttämättä ole parhaita mahdollisia.

Multimodaalisissa käyttöliittymissä modaliteettien soveltuminen tehtäväänsä on ensiarvoisen tärkeää. Tässä arvioinnissa puheen soveltuvuutta tekstieditorin ohjauksen ei arvioitu, mutta jatkotutkimuksen kannalta se olisi avainkysymyksiä. Modaliteettien soveltuminen tehtäviinsä on monimutkainen kysymys, koska vastaus riippuu hyvin monista asioista. Vastauksen hakemiseksi pitäisi ottaa huomioon ainakin sovelluksen tyyppi ja tarkoitus, sovelluksen käyttöympäristö, tiedonsiirtokanavan leveys ja stabiilius, oppimisesta aiheutuva vaiva, kohdeyleisö sekä modaliteettien ominaisuudet (Dybkjær et al. 2003).

8. YHTEENVETO

Tämän työn kohteena oli tutkia multimodaalisen synergian maksimointia käyttöliittymissä. Aihetta lähestyttiin tutkimalla puheohjauksen lisäämistä graafiseen käyttöliittymään. Työssä tarkasteltiin tätä pääosin puhekäyttöliittymien näkökulmasta, koska niitä ei ole vielä kattavasti tutkittu toisin kuin graafisia käyttöliittymiä. Näiden kahden modaliteetin on aiemmin havaittu tukevan kattavasti toisiaan.

Puheteknologiaa vaivaa vielä tänäkin päivänä tekniikan vajavaisuus. Puheentunnistusvirheitä ei voi välttää eikä täysin vapaata luonnollista kieltä voida ymmärtää. Tästä aiheutuu usein tekniikan määräävän puhekäyttöliittymien suunnittelua, jotta käyttäjän syötettä voidaan hallita ja virheitä välttää. Käyttäjän ja käytettävyyden kannalta tämä ei ole edullista ja on kenties suurin syy siihen, miksi puhekäyttöliittymät eivät ole saavuttaneet graafisten käyttöliittymien suosiota.

Työssä toteutettiin puheohjauksen lisääminen tekstieditoriin. Syntyneessä multimodaalisessa käyttöliittymässä puhetta käytettiin vain syötteen antoon ja järjestelmän palaute annettiin visuaalisessa muodossa. Tämä jako kuvastaa hyvin modaliteettien vahvuuksia, sillä puhe on hyvin ilmaisuvoimainen modaliteetti, jolla voidaan antaa monimutkaisia komentoja. Vastaavasti visuaalinen modaliteetti on parhaimmillaan palautteenannossa, sillä se on sekä pysyvää että rinnakkaista.

Puheohjaus oli jaettu kahteen moodiin: rajatun sanaston komentomoodiin ja vapaan sanaston sanelumoodiin. Komentomoodissa oli määriteltyinä vain tietty määrä avainlauseita, joilla tekstieditorin toimintoja ajettiin. Vastaavasti sanelumoodissa kielen käyttö oli vapaata ja puhe tulkittiin tekstieditoriin kirjoitettavaksi saneluksi. Havaintona ratkaisussa oli se, että puheohjauksen käyttö komentomoodissa oli helppoa ja tehokasta, koska tunnistustarkkuus oli hyvin korkea. Sanelumoodin tunnistustarkkuus puolestaan oli selvästi matalampi, joka kenties vieroittaa sen käyttöä. Tunnistustarkkuuden maksimoiminen onkin hyvin tärkeää tehokkaiden ja käytettävien käyttöliittymien suunnittelussa. Avaimena tarkkuuden parantamiseen voidaan pitää sanaston rajaamista mahdollisimman pieneksi.

Puheentunnistuksen virhealttiuden vuoksi virheiden korjaus nousee suureen osaan puhekäyttöliittymissä. Yksimodaalisessa puhesovelluksessa tämä on kenties raskasta ja vaikeaa, mutta multimodaalisessa käyttöliittymässä virheenkorjauksesta voidaan tehdä paljon tehokkaampaa. Graafisen puolen hyödyntäminen virheenkorjaukseen on ensiarvoisen hyödyllistä. Käyttäjät usein myös pyrkivät tehokkaimman modali-

teetin käyttöön ja vaihtavat modaliteettia virheiden sattuessa. Graafisen virheenkorjausmekanismin tarjoaminen onkin siten erittäin suositeltavaa multimodaalisissa käyttöliittymissä.

Puhesovellusten ja yleisesti multimodaalisten sovellusten käytettävyydessä voidaan noudattaa perinteisiä käytettävyyden lakeja. Lakien soveltaminen käyttötapaukseen voi olla haasteellista, koska vanhaa tietämystä asiasta ei ole kovin paljoa. Kuitenkin tutkimalla asiaa eteenpäin saadaan arvokasta käytännön kokemusta ja tasoitetaan tietä seuraaville sovelluskehittäjille.

Puhetta ja graafisia käyttöliittymiä yhdistävien sovellusten suunnitteluperiaatteet ovat vielä tutkimuksen alla. Vielä ei ole selvää kuinka parhaiten hyödyntää modaliteetteja toisiaan tukevalla tavalla. Asiasta saadaan lisätietoa vain kun näitä modaliteetteja yhdistävien sovellusten määrä kasvaa ja tutkimus lisääntyy käytännön sovellusten ja käyttäjätutkimusten kautta. Varmaa on vain se, että multimodaaliset sovellukset ovat yksimodaalisia tehokkaampia, käytettävämpiä ja joustavampia.

LÄHTEET

J. Bradford. The human factors of speech-based interfaces: A research agenda. *ACM SIGCHI Bulletin*, 27(2):61–67, 1995.

A. Chapanis. Interactive human communication. *Scientific American*, 232:36–42, 1975.

F. Chen. *Designing Human Interface in Speech Technology.* Springer, New York, 2006, 382 s.

P. R. Cohen, S. L. Oviatt. The role of voice input for human machine communication. *Voice Communication Between Humans and Machines*, s. 34–75, Washington D.C., 1994. National Academy Press.

L. Dybkjær, N. O. Bernsen. Usability evaluation in spoken language dialogue systems. *Proceedings of the Workshop on Evaluation for Language and Dialogue Systems, Association for Computational Linguistics 39th Annual Meeting and 10th Conference of the European Chapter (acl/eacl) 2001*, s. 9–18, 2001.

L. Dybkjær, N. O. Bernsen, W. Minker. Evaluation and usability of multimodal spoken language dialogue systems. *Speech Communication*, s. 33–54, 2003.

V. Farinazzo, M. Salvador, A. L. S. Kawamoto, J. S. de Oliveira Neto. An Empirical Approach for the Evaluation of Voice User Interfaces. *User Interfaces*, s. 153–164. InTech, 2010.

M. A. Grasso, D. Ebert, T. Finin. The integrality of speech in multimodal interfaces. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 5(4): 303–325, 1998.

J. Gustafson. *Developing Multimodal Spoken Dialogue Systems. Empirical Studies of Spoken Human-Computer Interaction.* Doctoral dissertation, KTH. Department of Speech, Music and Hearing, Stockholm, 2002, 96 s.

D. Jones, C. R. Frankish, K. Hapeshi. Automatic speech recognition in practice. *Behaviour & Information Technology*, 11(2):109–122, 1992.

C. Kamm. User interfaces for voice applications. *Voice Communication Between Humans and Machines*, s. 422–442, Washington D.C., 1994. National Academy Press.

M. Kurimo. Puheentunnistus. *Puhe ja kieli*, 28(2):73–83, 2008.

J. Lai, N. Yankelovich. Conversational speech interfaces. *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*, s. 698–713. Lawrence Erlbaum Associates, Inc., New Jersey, USA, 2003.

J. Makhoul, R. Schwartz. State of the art in continuous speech recognition. *Voice Communication Between Humans and Machines*, s. 165–198, Washington D.C., 1994. National Academy Press.

A. Mane, S. Boyce, D. Karis, N. Yankelovich. Designing the user interface for speech recognition applications. *SIGCHI Bulletin*, 28(4):29–34, 1996.

J. Nielsen. *Usability Engineering.* Academic Press, San Francisco, 1993.

S. Oviatt. Mutual disambiguation of recognition errors. *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, s. 576–583, 1999a.

S. Oviatt. Ten myths of multimodal interaction. *Communications of the ACM*, 42 (11):74–81, 1999b.

S. Oviatt, R. VanGent. Error resolution during multimodal human-compute interaction. *Proceedings of the International Conference on Spoken Language Processing*, s. 204–207, Japan, 1994.

S. Oviatt, G. A. Levow, M. Maceachern, K. Kuhn. Predicting hyperarticulate speech during human-computer error resolution. *Speech Communication*, 24:87–110, 1998.

S. Oviatt, R. Coulston, S. Tomko, B. Xiao, R. Lunsford, M. Wesson, L. Carmichael. Toward a theory of organized multimodal integration patterns during human-computer interaction. *Proceedings of the 5th international conference on Multimodal interfaces*, s. 44–51, 2003.

M. Perakakis. *Blending Speech and Graphical User Interfaces. An empirical study on multimodal mobile interaction.* PhD thesis, Technical University Crete. Department of Electronics and Computer Engineering, Chania, Greece, 2011, 143 s.

V. F. M. Salvador, J. S. de Oliveira Neto, A. S. Kawamoto. Requirement engineering contributions to voice user interface. *Proceedings of the First International Conference on Advances in Computer-Human Interaction*, s. 309–314, Washington, DC, USA, 2008. IEEE Computer Society.

J. Schalkwyk, D. Beeferman, F. Beaufays, B. Byrne, C. Chelba, M. Cohen, M. Kamvar, B. Strope. "Your Word is my Command": Google Search by Voice: A Case Study. *Advances In Speech Recognition: Mobile Environments, Call Centers and Clinics*, s. 61–90. Springer, 2010.

C. Schmandt. *Voice Communication with Computers: Conversational Systems*. Van Nostrand Reinhold, New York, 1994, 343 s.

B. Schneiderman. *Designing the User Interface: Strategies For Effective Human-Computer Interaction*. Addison-Wesley, 1992, 573 s.

B. Suhm, B. Myers, A. Waibel. Multimodal error correction for speech user interfaces. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 9(1):60–98, 2001.

M. Turunen. Puheohjaus 3D-käyttöliittymissä. Pro Gradu-tutkielma, Tampereen yliopisto, tietojenkäsittelyopin laitos, Tampere, Suomi, 1998.

W3C. Multimodal interaction requirements. [WWW]. [viitattu 20.12.2012]. Saatavissa: `http://www.w3.org/TR/mmi-reqs/`.

wbond. Sublime Text Packages. [WWW]. [viitattu 16.3.2013]. Saatavissa: `http://wbond.net/sublime_packages/community`.